An Interleaved Multi-mode \( \Delta \Sigma \) RF-DAC with Fully Integrated, AC Coupled Digital Input

Dissertation

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

By

Jamin J. McCue, B.S.E.E., M.S.

Graduate Program in Electrical and Computer Engineering

The Ohio State University

2015

Dissertation Committee:

Dr. Waleed Khalil, Advisor
Dr. Patrick Roblin
Dr. Steven Bibyk
Abstract

With the continued growth of mobile communications, large portions of the RF spectrum are being utilized for a variety of wireless applications. For next-generation systems to adapt to this crowded and fluctuating wireless environment, future radio hardware must be capable of flexible and reconfigurable operation. Key to this endeavor is the development of high-performance digital-to-analog converters which can directly synthesize RF signals, bypassing the need for analog up-conversion and other RF processing by pushing functionality into the digital domain.

In this work, a multi-mode delta-sigma (∆Σ) RF digital-to-analog converter (RF-DAC) is developed for direct digital-to-RF synthesis. The proposed architecture uses only a single clock frequency ($f_S$) for RF generation and includes a reconfigurable ∆Σ modulator (DSM) that operates in band-pass (BP) and high-pass (HP) modes to synthesize signals around $f_S/4$, $f_S/2$, or $3f_S/4$. Analog interleaving via two 3-bit DACs is used to reject the first DAC image, simultaneously doubling the usable bandwidth of the HP DSM, increasing the SNR, and easing filtering requirements. After a theoretical discussion, the proposed architecture is demonstrated by an initial prototype implemented in a 130 nm SiGe BiCMOS process and operating at $f_S = 2GH\text{z}$. The design realizes a signal-to-image rejection ratio (SIRR) of 72 dB, an SNR of 54.5 dB over a 50 MHz bandwidth, and an in-band SFDR of 58.5 dB.
In a second revision, an on-chip 14-bit DSM is included, implemented as an array of pipelined 1-bit pipelined subtracters to generate 3-bit, $f_s$-rate input data. The second prototype also utilizes on-chip amplitude and timing calibration along with a CML data path to improve DAC speed, linearity, and SNR. Measurements at $f_s = 2\, \text{GHz}$ yield a $76.2\,\text{dB}$ SIRR, $76.2\,\text{dB}$ SFDR over a $100\,\text{MHz}$ bandwidth, $-80\,\text{dBc}$ IM3, $-67.2\,\text{dB}$ WCDMA ACLR and $-66.4\,\text{dBc}$ LTE ACLR.

To enable the high-speed data input required by the RF-DAC, a fully integrated, AC coupled pulse receiver is designed in support of this work. The low power and area efficient receiver is implemented as a fully integrated pulse receiver, eliminating the need for large, board-mounted capacitors. Additionally, the pulse receiver topology minimizes baseline wander with return-to-zero (RZ) signaling and removes the need for data encoding by internally latching long strings of continuous data. A common mode feedback circuit employs replica biasing to ensure operation over PVT variations and across various modes of operation. In a low and high power mode, the proposed receiver is tested up to $7.5\,\text{Gb/s}$ and $10\,\text{Gb/s}$, respectively, achieving a peak operation efficiency of $0.54\,\text{mW/Gb/s}$ and a BER $< 10^{-13}$.

Altogether, this work describes and validates a novel method of direct digital-to-RF synthesis. The design is traced from conceptual analysis through implementation and testing. Other circuits, such as the LVDS pulse receiver, are designed in support of this work and serve as an indicator of both the design challenges and opportunities found at the interface between the digital and RF domains.
To my wife Andrea: my truest critic and my closest friend.

“...for a crowd is not company, and faces are but a gallery of pictures, and talk but a tinkling cymbal, where there is no love. [Life] is a mere and miserable solitude to want true friends, without which the world is but a wilderness.” - F. Bacon

Soli Deo Gloria
I would like to thank my family, especially my parents, for their unwavering support throughout my education, and indeed my life. I am sure that I am a direct product of their hard-work and intentional upbringing, apart from which I would not be who I am.

To my friends in Columbus, all of you have been more than just a needed distraction from school; you have shaped me personally more than you know.

I am indebted to the members of my lab for their technical assistance and profitable discussion, without which this work would not be what it is. Additional thanks to Samantha McDonnell for the design and use of several circuits in this work. Further, I must thank Lucas Duncan for his contributions to this work specifically and for his friendship in general. Because of him I never lacked an honest opinion or needed distraction.

I would like to thank the members of the RYDI team at the AFRL (Brad, Greg, Paul, Tony, Len, Aji, Vipul, and Cari) for their assistance financial, technical, and otherwise. Surely I have gained much more from their input than I have given in return, and I am forever grateful for the opportunities they have afforded me.

I would like to thank Dr. Brian Dupaix for his trail-blazing work in the CLASS lab. Without his guidance and hard work over many years, the infrastructure needed
for this project would not have been available. In addition, his many directives and urgings served to mold me into a better engineer, for which I am in his debt.

Thanks to Dr. James Wilson and Vipul Patel for their assistance with the receiver portion of this work.

Many thanks to my committee members, Dr. Steven Bibyk and Dr. Patrick Roblin, 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 my potential when I knew next to nothing. This work is a direct result of his teaching, oversight, and guidance. Over last several years at OSU, the example of his tireless work and technical depth have and will continue to influence me.

To those I have forgotten, for I am sure there are a few, your contributions to this work, whether direct or indirect, deserve more than a mention on this page...

-Thank you.
Vita

2009 ................................. B.S.E.E., Cedarville University

2009 ................................. Student Research Associate,
Center for Directed Energy,
Air Force Institute of Technology

2010-2011 .............................. AFRL Sensors Fellowship

2014 ................................. M.S., The Ohio State University

2009-present ........................ Graduate Research Associate,
The Ohio State University

Publications

Research Publications

Tahir Hussain, James Li, Ken Elliott, Donald Hitko, Paul Watson, Jamin McCue, Kristian Madsen, Heterogeneous Integration: Microsystems on a Chip Through an Optimal Combination of Diverse Electronic Micro-Components (Invited), International Symposium on Compound Semiconductors (ISCS), June 2015


Jamin J. McCue, Matthew Casto, James C. Li, Paul Watson, Waleed Khalil, An Active Double-Balanced Down-Conversion Mixer in InP/Si BiCMOS Operating from 70-110 GHz, Compound Semiconductor Integrated Circuit Symposium (CSICS), Oct. 2014 (Nominated for Best Student Paper)


Qiyang Wu; Elabd, S.; **McCue, J.J.**; Khalil, W., "Analytical and Experimental Study of Tuning Range Limitation in mm-Wave CMOS LC-VCOs," International Symposium on Circuits and Systems (ISCAS), May 2013


Qiyang Wu, Salma Elabd, **Jamin J. McCue** and Waleed Khalil, A 10mW 37.8GHz Current Redistribution BiCMOS VCO with an Average FOMT of -193.5dBc/Hz, International Solid-State Circuits Conference (ISSCC), 2013.

Wu, Qiyang; Quach, Tony; Mattamana, Aji; Elabd, Salma; Dooley, Steven R.; **McCue, Jamin J.**; Orlando, Pompei L.; Creech, Gregory L.; Khalil, Waleed; , "Design of Wide Tuning-Range mm-Wave VCOs Using Negative Capacitance," Compound Semiconductor Integrated Circuit Symposium (CSICS), Oct. 2012


Fields of Study

Major Field: Electrical and Computer Engineering

Specialization: Analog and RF Electronics
Table of Contents

Abstract ........................................................................................................... ii

Dedication ......................................................................................................... iv

Acknowledgments ............................................................................................ v

Vita ...................................................................................................................... vii

List of Tables .................................................................................................... xiii

List of Figures ................................................................................................... xvi

1. Introduction ................................................................................................... 1
   1.1 Analog Transmitter Architectures ......................................................... 1
   1.2 A Digital Alternative: The RF-DAC .................................................... 2
      1.2.1 Nyquist Rate RF-DACs ................................................................. 3
      1.2.2 ΔΣ RF-DACs ............................................................................... 6
      1.2.3 Direct Synthesis RF-DACs ............................................................ 8
      1.2.4 The Proposed Multi-mode, Interleaved ΔΣ DAC ....................... 9
   1.3 High Speed Data I/O ............................................................................. 10
      1.3.1 AC-Coupled Serial I/O ................................................................. 10
      1.3.2 The Proposed AC-Coupled Receiver ......................................... 14
   1.4 Outline .................................................................................................... 15

2. AC Coupled Data Input ................................................................................. 16
   2.1 Pulse Receiver ....................................................................................... 18
      2.1.1 Latching ...................................................................................... 21
   2.2 The Bias Latched Pulse Receiver .......................................................... 22
      2.2.1 Input Amplifier ............................................................................ 23
### 2.2 Level Shifting and CMFB
- 2.2.2 Level Shifting and CMFB ........................................ 25
- 2.2.3 Simulation ......................................................... 26

### 2.3 LVDS Receiver Testing
- 2.3.1 Measurement Results ................................................ 30
- 2.3.2 Comparison ......................................................... 33

### 3. The Interleaved, Multi-mode ΔΣ DAC
- 3.1 Reconfigurable ΔΣ Modulation ........................................ 34
  - 3.1.1 Digital Quantizer ................................................. 36
  - 3.1.2 Feedback Filter .................................................. 38
  - 3.1.3 Input Summation ................................................ 43
- 3.2 Multi-Nyquist Operation ............................................... 44
  - 3.2.1 Interleaving ...................................................... 45
  - 3.2.2 Inter-DAC Amplitude and Timing Errors ......................... 50
  - 3.2.3 Intra-DAC Errors ............................................... 54
- 3.3 Comparison to Nyquist-Rate Architecture .......................... 55
  - 3.3.1 In-band SNR and ENOB ......................................... 55
  - 3.3.2 Noise Floor .................................................... 59
  - 3.3.3 ΔΣ Output Efficiency .......................................... 61

### 4. ΔΣ DAC - First Revision
- 4.1 Circuit Design ..................................................... 65
  - 4.1.1 DAC and Data Retiming ......................................... 65
  - 4.1.2 Data I/O and Loopback Testing ................................ 68
- 4.2 Measurement .......................................................... 68
  - 4.2.1 Test Infrastructure ............................................. 69
  - 4.2.2 Results ......................................................... 71
  - 4.2.3 Take Aways .................................................... 74

### 5. ΔΣ DAC - Second Revision
- 5.1 Circuit Design ..................................................... 77
  - 5.1.1 Serial Data Input ................................................ 77
  - 5.1.2 ΔΣ Modulator ................................................... 78
  - 5.1.3 Data Path and DAC ............................................ 83
- 5.2 Measurement .......................................................... 89
  - 5.2.1 Test Infrastructure ............................................. 89
  - 5.2.2 Results ......................................................... 92
  - 5.2.3 Comparison .................................................... 98
6. Conclusion and Future Work ........................................ 100
   6.1 Work Summary and Conclusion ............................... 100
   6.2 Future Work .................................................. 102
       6.2.1 Scaling Improvements ................................. 102
       6.2.2 System Integration ................................. 103
   6.3 Final Thoughts ............................................. 106

Appendices ..................................................... 107

A. ∆Σ DAC Model ................................................. 107
B. ∆Σ DAC Random Jitter Model ................................. 115
C. Testing Methodology - Rev. 2 ................................ 127
    C.1 Test Equipment ............................................ 127
    C.2 Performance and Verification Tests ..................... 128
D. SPI Register Definitions .................................... 133
    D.1 Register Map .............................................. 133
    D.2 Register Definitions .................................. 134

Bibliography .................................................... 154

xii
List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Receiver Circuit Sizing</td>
<td>23</td>
</tr>
<tr>
<td>2.2</td>
<td>Receiver Power Settings and Performance</td>
<td>31</td>
</tr>
<tr>
<td>2.3</td>
<td>Comparison of Gb/s Pulse Receivers</td>
<td>33</td>
</tr>
<tr>
<td>5.1</td>
<td>Power Usage</td>
<td>98</td>
</tr>
<tr>
<td>5.2</td>
<td>$\Delta \Sigma$ RF-DAC Measurement Comparison</td>
<td>98</td>
</tr>
<tr>
<td>C.1</td>
<td>Board Supply Voltages</td>
<td>129</td>
</tr>
<tr>
<td>D.1</td>
<td>Register Map for DAC Serial Interface</td>
<td>133</td>
</tr>
<tr>
<td>D.2</td>
<td>DAC_CTL0 Register Description</td>
<td>134</td>
</tr>
<tr>
<td>D.3</td>
<td>DAC_CTL1 Register Description</td>
<td>135</td>
</tr>
<tr>
<td>D.4</td>
<td>DAC_STATUS Register Description</td>
<td>136</td>
</tr>
<tr>
<td>D.5</td>
<td>LOOPBACKx Register Description</td>
<td>136</td>
</tr>
<tr>
<td>D.6</td>
<td>DATA_RX_ADJx Register Description</td>
<td>137</td>
</tr>
<tr>
<td>D.7</td>
<td>CAL_CUT_L Register Description</td>
<td>137</td>
</tr>
<tr>
<td>D.8</td>
<td>CAL_CUT_S Register Description</td>
<td>138</td>
</tr>
<tr>
<td>D.9</td>
<td>CAL_LO_ADJ Register Description</td>
<td>139</td>
</tr>
</tbody>
</table>
D.10 CAL_LO_DATA_L Register Description .................................. 139
D.11 CAL_LO_DATA_S Register Description .................................. 140
D.12 CAL_MEAS_DELAY Register Description ................................. 140
D.13 CAL_OS_L Register Description ........................................... 141
D.14 CAL_OS_S Register Description ........................................... 141
D.15 CAL_GE_L Register Description ........................................... 142
D.16 CAL_GE_S Register Description ........................................... 142
D.17 CAL_TARGET_A Register Description ................................... 143
D.18 CAL_TARGET_T Register Description ................................... 143
D.19 CAL_I_RESx Register Description ........................................ 144
D.20 CAL_T_RESx Register Description ........................................ 144
D.21 CAL_DATA_SEL0 Register Description .................................. 145
D.22 CAL_DAC_DATA_SEL0 Register Description ............................. 146
D.23 CAL_TADJ_DATA_SEL0 Register Description ........................... 147
D.24 CAL_DATAx_L Register Description ....................................... 147
D.25 CAL_DATAx_S Register Description ....................................... 148
D.26 CAL_DAC_DATAx_L Register Description ................................ 148
D.27 CAL_DAC_DATAx_S Register Description ................................ 149
D.28 CAL_TADJ_DATAx_L Register Description ............................... 150
D.29 CAL_TADJ_DATAx_S Register Description ............................... 150
D.30 DS_CTL Register Description .............................................. 151
D.31 CLK_CTLx Register Description ........................................ 151
D.32 DS_OFFSETx Register Description ........................................ 152
D.33 CLK_ALIGN Register Description ........................................ 153
D.34 DATA_ARR Register Description ........................................ 153
## List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>The Progression of Software Defined Transmitters</td>
<td>3</td>
</tr>
<tr>
<td>1.2</td>
<td>Mixing DAC Topologies</td>
<td>5</td>
</tr>
<tr>
<td>1.3</td>
<td>AC Coupled Receivers</td>
<td>12</td>
</tr>
<tr>
<td>1.4</td>
<td>System Overview</td>
<td>15</td>
</tr>
<tr>
<td>2.1</td>
<td>Unencoded AC Coupled Input</td>
<td>17</td>
</tr>
<tr>
<td>2.2</td>
<td>AC Coupled Pulse Transient</td>
<td>18</td>
</tr>
<tr>
<td>2.3</td>
<td>Capacitor Sizing</td>
<td>20</td>
</tr>
<tr>
<td>2.4</td>
<td>Bias Latched AC Coupling</td>
<td>22</td>
</tr>
<tr>
<td>2.5</td>
<td>LVDS Pulse Receiver Block Diagram</td>
<td>23</td>
</tr>
<tr>
<td>2.6</td>
<td>Receiver Schematic</td>
<td>24</td>
</tr>
<tr>
<td>2.7</td>
<td>Coupled Input Simulation</td>
<td>25</td>
</tr>
<tr>
<td>2.8</td>
<td>Receiver Level Shift</td>
<td>26</td>
</tr>
<tr>
<td>2.9</td>
<td>Output Eye Simulation</td>
<td>27</td>
</tr>
<tr>
<td>2.10</td>
<td>Receiver Test Chip</td>
<td>29</td>
</tr>
<tr>
<td>2.11</td>
<td>LVDS Pulse Receiver Test Board</td>
<td>29</td>
</tr>
<tr>
<td>Section</td>
<td>Page</td>
<td></td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>------</td>
<td></td>
</tr>
<tr>
<td>5.13 On-chip DSM Measured Spectrum</td>
<td>92</td>
<td></td>
</tr>
<tr>
<td>5.14 Measured HP and BP Spectrums</td>
<td>93</td>
<td></td>
</tr>
<tr>
<td>5.15 Measured 800 MHz Spectrum</td>
<td>94</td>
<td></td>
</tr>
<tr>
<td>5.16 Measurement Results Across Sample Rate</td>
<td>95</td>
<td></td>
</tr>
<tr>
<td>5.17 WCDMA Measurement Results</td>
<td>96</td>
<td></td>
</tr>
<tr>
<td>5.18 LTE Measurement Results</td>
<td>97</td>
<td></td>
</tr>
<tr>
<td>6.1 Advantages of Scaling</td>
<td>103</td>
<td></td>
</tr>
<tr>
<td>6.2 Future System Integration</td>
<td>105</td>
<td></td>
</tr>
<tr>
<td>A.1 Simulink DAC Model</td>
<td>114</td>
<td></td>
</tr>
<tr>
<td>C.1 Test Setup</td>
<td>128</td>
<td></td>
</tr>
<tr>
<td>C.2 Test Board</td>
<td>129</td>
<td></td>
</tr>
<tr>
<td>C.3 Clock and Loopback Testing</td>
<td>130</td>
<td></td>
</tr>
<tr>
<td>C.4 Calibration</td>
<td>131</td>
<td></td>
</tr>
<tr>
<td>C.5 Full Testing</td>
<td>132</td>
<td></td>
</tr>
<tr>
<td>D.1 DAC_CTL0 Register Fields</td>
<td>134</td>
<td></td>
</tr>
<tr>
<td>D.2 DAC_CTL1 Register Fields</td>
<td>135</td>
<td></td>
</tr>
<tr>
<td>D.3 DAC_STATUS Register Fields</td>
<td>135</td>
<td></td>
</tr>
<tr>
<td>D.4 LOOPBACKx Register Fields</td>
<td>136</td>
<td></td>
</tr>
<tr>
<td>D.5 DATA_RX_ADJx Register Fields</td>
<td>137</td>
<td></td>
</tr>
<tr>
<td>D.6 CAL_CUT_L Register Fields</td>
<td>137</td>
<td></td>
</tr>
<tr>
<td>D.7 CAL_CUT_S Register Fields</td>
<td>138</td>
<td></td>
</tr>
</tbody>
</table>
D.8 CAL_LO_ADJ Register Fields ........................................ 138
D.9 CAL_LO_DATA_L Register Fields .................................... 139
D.10 CAL_LO_DATA_S Register Fields .................................. 139
D.11 CAL_MEAS_DELAY Register Fields ................................. 140
D.12 CAL_OS_L Register Fields .......................................... 141
D.13 CAL_OS_S Register Fields .......................................... 141
D.14 CAL_GE_L Register Fields .......................................... 142
D.15 CAL_GE_S Register Fields .......................................... 142
D.16 CAL_TARGET_A Register Fields .................................. 143
D.17 CAL_TARGET_T Register Fields .................................. 143
D.18 CAL_I_RESx Register Fields ....................................... 144
D.19 CAL_T_RESx Register Fields ....................................... 144
D.20 CAL_DATA_SEL0 Register Fields ................................. 145
D.21 CAL_DAC_DATA_SEL0 Register Fields ......................... 146
D.22 CAL_TADJ_DATA_SEL0 Register Fields ......................... 146
D.23 CAL_DATAx_L Register Fields .................................... 147
D.24 CAL_DATAx_S Register Fields .................................... 148
D.25 CAL_DAC_DATAx_L Register Fields ............................ 148
D.26 CAL_DAC_DATAx_S Register Fields ............................ 149
D.27 CAL_TADJ_DATAx_L Register Fields ............................ 149
D.28 CAL_TADJ_DATAx_S Register Fields . . . . . . . . . . . . . . . . 150
D.29 DS_CTL Register Fields . . . . . . . . . . . . . . . . . . . . . . 150
D.30 CLK_CTLx Register Fields . . . . . . . . . . . . . . . . . . . . . 151
D.31 DS_OFFSETx Register Fields . . . . . . . . . . . . . . . . . . . . 152
D.32 CLK_ALIGN Register Fields . . . . . . . . . . . . . . . . . . . . . 152
D.33 DATA_ARR Register Fields . . . . . . . . . . . . . . . . . . . . . 153
Chapter 1: Introduction

The last decade has seen unparalleled growth in the bandwidths available over mobile wireless connections. Broad use of the RF spectrum has pushed transceivers to operate in more crowded wireless environments while also requiring substantial increases in data rates with each generation of transceiver. This crowded and rapidly changing application space demands radios which are flexible with respect to carrier frequency and adaptable with respect to modulation scheme and bandwidth. The term Software Defined Radio (SDR) has come to encapsulate the end goal of this evolution — complete flexibility and adaptability through the programming of a highly configurable radio [1, 2]. Implicit in this definition is the use of multiple, simultaneous frequency bands and modulations schemes within a single radio, a truly universal transceiver with the benefits of scalability across technology and resistance to variations in process and environment.

1.1 Analog Transmitter Architectures

Single-band Homodyne and Heterodyne transmitters have long served as the default architecture for many wireless applications due to their reliability and intuitive design [3–5]. In these architectures, a digital baseband signal is passed through a low-frequency digital-to-analog converter (DAC) before being up-converted to an RF
band in one (Homodyne) or multiple (Heterodyne) steps. After analog processing, the signal is amplified and transmitted via a power amplifier and antenna. These conventional designs, however, do not lend themselves to the spectrum agility required by modern mobile communications. Due to their rigid and analog-centric design, there is an inherent trade-off made between frequency tuning and performance [6, 7]. This is illustrated in Fig. 1.1(a) where the two architectures are depicted as low-flexibility designs, highlighting some of the difficulties of reconfigurable implementation such as increased LO phase noise and poor filter Q-factor. To meet the requirements of future SDRs, highly digital transmitters have been proposed which take advantage of adaptive hardware and process scaling to improve flexibility and performance. As illustrated in Fig. 1.1(b), high-speed DACs play a central role in the implementation of these digital topologies by extending the digital domain far into the transmit chain. However, placing DACs in the RF signal path requires large increases in sample rate while maintaining linear performance. This new paradigm has spurred significant research into high-frequency DACs capable of high-fidelity RF output: the RF-DAC.

1.2 A Digital Alternative: The RF-DAC

Over the last decade, advances in digital and RF integration have led to the development of RF-DACs (or mixing-DACs) capable of low noise and high linearity direct digital-to-RF synthesis [8]. Unlike traditional D/A conversion where the output signal is restricted to less than half of the DAC sample rate, mixing-DACs (Fig. 1.2(a)) decouple the sample-rate and output frequency by combining the up-conversion mixer and DAC into a single, typically current-mode design capable of low-distortion RF signal generation via weighted current sources mixed by a local oscillator (LO) [7, 9].
Due to the high output frequencies of these designs, the D/A conversion can be located further down the transmit chain, removing the need for a static analog up-conversion stage and paving the way for increasingly digital transmitter topologies.

1.2.1 Nyquist Rate RF-DACs

Despite these advances, the low resolutions of early mixing-DAC designs did not meet the noise floor specifications of current communications standards [10]. More recently, architectures have sought to satisfy these stringent noise requirements through both higher resolution (increased $N_N$) and increased sample rate ($f_{SN}$). In [11], an oversampling 14-bit polar mixing-DAC is designed for low far-out noise, critical for 2G/3G communications, while [12] increases resolution further, implementing a highly linear 16-bit mixing-DAC, sampled in the GS/s range and tunable across multiple GHz.
However, to realize these high resolution topologies, dozens of DAC cells are required, dramatically increasing the analog and digital footprint of the design. Due to this expanded footprint, the amplitude and timing mismatch among the DAC cells is increased, degrading the DAC linearity and introducing data dependent distortion. Moreover, DAC current mismatch is exacerbated by reducing the size of the cell current source, limiting attempts to shrink the large DAC core. To minimize the impact of these mismatches, calibration and dynamic element matching (DEM) techniques have been proposed, correcting [13, 14], sorting-and-combining [15], or randomizing [16] the error between DAC cells. However, even with these techniques, a large number of cells are still needed to achieve the desired noise floor, increasing the area and complexity of the design as well as adding long calibration time during in-situ operation. In addition to the costs of calibration, the large number of DAC cells limit the high frequency performance of a design by loading the DAC output node and modulating the DAC output impedance [17].
Figure 1.2: An IQ mixing-DAC showing both (a) Nyquist-rate and (b) oversampled varieties, where the LO requires wide tuning for multi-band operation. (c) The interleaved, reconfigurable ΔΣ modulator and DAC showing multi-Nyquist RF synthesis.
1.2.2 ΔΣ RF-DACs

As an alternative to high-resolution, low-noise mixing-DACs, ΔΣ mixing architectures (Fig. 1.2(b)) significantly reduce the size of the DAC with a preceding ΔΣ modulator (DSM). The DSM shapes excess quantization noise away from an RF passband, allowing low-noise signal synthesis within a narrow bandwidth. The mixing DAC is then implemented with minimal resolution (1-3 bits), thereby leveraging scalable digital processing to reduce analog complexity [18]. Furthermore, the low resolution DAC core reduces output loading and internal mismatch while minimizing the overhead of calibration circuitry and limiting the time needed to run a given calibration technique.

These advantages are first described in [19] where the performance of a conventional ΔΣ DAC is compared to that of a ΔΣ mixing architecture. Pre-computed ΔΣ data is fed into an eight-cell DAC, enabling this initial development of the mixing-DAC architecture and showcasing the potential of low-resolution design for RF synthesis.

The subsequent work in [20] digitally up-converts a lowpass (LP) DSM output to an intermediate frequency (IF), implementing a homodyne architecture spread across the digital and RF domains. This upconversion method offsets the signal bandwidth from the DAC LO, clearing the output band of unwanted LO leakage and mixing images which result from I/Q mismatch. The ΔΣ mixing DAC uses an off-chip analog LO to drive 3-bit I and Q DACs. A self-tuned LC filter is constructed at the DAC output to eliminate out of band noise for signals around 5 GHz. However, wide reconfigurability is restricted by the need for wide LO and filter tuning in the design.
Another digital IF architecture, reported in [21], implements a band-pass (BP) DSM at IF, creating a low noise passband which is subsequently up-converted to RF via an inherently-linear, 1-bit mixing-DAC. Additionally, the implemented DAC is cascaded to realize a semi-digital reconstruction filter within the design. This novel use of the domain interface ideally leverages the timing of the digital, the simple summation of the RF, and the small size of the DAC to realize a digitally reconfigurable integrated filter. The 1-bit architecture improves IM3, but the low baseband sampling frequency limits its noise and bandwidth performance.

In [22], a ∆Σ topology enables digital I/Q mixing beyond IF by directly up-converting high-speed DSM data with an all-digital mixer, the output of which is fed into weighted I and Q current cells for dynamic power control. This design bridges the ∆Σ and Nyquist domains by implementing a moderate resolution (> 6 bit) DAC which utilizes noise shaping in the pursuit of low noise RF performance.

Despite the advantages of ∆Σ modulation, the mixing-DAC architecture has several inherent limitations which impair their reconfigurable operation. First, the absence of a reconstruction filter before up-conversion gives rise to close-to-carrier images in the DAC output. Even with the attenuation of hold shaping [23], mixing-DACs have shown a maximum image suppression of 54 dB [24]. Instead, designs sample the baseband signal at frequencies which approach or exceed the LO frequency [20, 22], pushing DAC images further out of band. However, these high frequency sampling clocks must be accurately aligned to the LO signal to mask the jitter of incoming data and maintain in-band noise performance [21]. Additionally, since I/Q mixing and combining is performed in the analog domain, both amplitude and timing mismatches can corrupt the output error vector magnitude and induce mixing images.
This is further exacerbated by the challenge of multi-band operation where the LO must provide both wide frequency tuning and low phase noise.

1.2.3 Direct Synthesis RF-DACs

This is in contrast to fully digital mixing which does not introduce IQ mismatch nor does it require the alignment of a widely tuned analog LO [25]. Instead, up-sampled data is multiplied by a clock signal to digitally synthesize RF waveforms. While the output frequency is limited by the digital sample rate of the system (encouraging architectures which utilize easy-to-realize multiplication), the speed and performance of digital mixing will continue to improve with technology scaling. Therefore, a viable alternative to the mixing-DAC architecture becomes that of fully-digital I/Q mixing coupled with a high-speed ∆Σ DAC.

Such a system is described in [26] where a single, high-speed clock is used for both a LP DSM and to directly synthesize the results of all-digital mixing and I/Q combining. The baseband signal is first up-sampled to the rate of the clock (created from an on-chip delay locked loop) and then passed through the DSM. The shaped passband is up-converted to RF by a high-speed mux, toggling between inverted and non-inverted I/Q data. At the output of the design, [26] employs an inherently-linear 1-bit inverter to directly synthesize a GHz-range RF output by leveraging the first image of the sampled waveform. However, due to its supply noise sensitivity, the single-bit inverter topology contributes high jitter and, as a result, increases in-band noise. Additionally, when utilizing outputs in the second Nyquist zone, a high power image is present in the first, requiring high stopband attenuation in a subsequent reconstruction filter. In this design, the viability of a fully digital architecture is
demonstrated, but the RF performance must be improved before integration into a wireless system.

1.2.4 The Proposed Multi-mode, Interleaved $\Delta\Sigma$ DAC

In this work, a multi-order, multi-band $\Delta\Sigma$ DAC is proposed to leverage the advantages of all-digital I/Q up-conversion while also improving jitter performance and canceling unwanted, high-power image replicas. This is accomplished by arraying two time-interleaved channels containing both the DSM and DAC while utilizing readily available $180^\circ$ clocks. When summed, the channel outputs cancel the unwanted images and reject their associated non-linearity spurs, leveraging multiple Nyquist zones to achieve high-frequency RF signal synthesis [27]. By canceling the first DAC image, the proposed design uniquely enables high-pass (HP) $\Delta\Sigma$ modulation which, as can be seen in Fig. 1.2(c), has twice the bandwidth (red) of BP $\Delta\Sigma$ modulation (grey) due to its near Nyquist operation [28]. Additionally, any residual DAC image is located in-band, acting as a self-interfering signal, as opposed to an out-of-band spurious emission, limiting potential interference with adjacent communications. It is worth noting that including the DAC as part of the interleaved data channel minimizes the speed and timing constraints compared with other interleaving schemes which focus exclusively on the DSM and require a high-speed multiplexer and DAC [29, 30]. Taking advantage of the all-digital DSM, a reconfigurable modulator is created by switching between various MASH stages and filter functions, synthesizing RF passbands at 1/4, 1/2, or 3/4 of the system clock rate.
Altogether, the proposed architecture is able to leverage parallel, high-speed DACs to directly output RF signals, and ΔΣ modulation is used to minimize the degradation of the DAC performance at high frequency.

1.3 High Speed Data I/O

A significant issue in the implementation and testing of the proposed DAC, especially given its high input resolution and interleaved topology, is high-speed data input. While baseband DACs need only receive a data stream capable of synthesizing the signal bandwidth (a few hundred MHz at most), the proposed DAC directly synthesizes modulated RF signals which require high-rate data containing both the desired signal and an RF carrier. Methods of data I/O such as multiplexing lower bit-rate data and on-chip memory have been previously employed to meet data throughput requirements in DACs [31–33], sidestepping the complexities of high-speed, chip-to-chip serial communications by sacrificing chip area, power, time, or additional I/O pins. Instead, a more direct approach is taken in this work, that of real-time data input [34, 35]. To this end, current trends in high-performance serial I/O are discussed and a novel, fully integrated receiver is presented as a means of enabling the real-time operation of the interleaved DAC architecture.

1.3.1 AC-Coupled Serial I/O

With the on-going progress of integrated circuit scaling, on-chip communications see continued improvement due to reductions in loading and circuit spacing. Chip-to-chip data links, however, do not experience these same benefits, relying instead on new and increasingly complex transceiver topologies to overcome package and board-level limitations. In pursuit of higher data rates and increased power and area
efficiency, high-performance chip-to-chip communications have evolved from highly-parallel topologies through DC-coupled serial designs [36] with recent architectures now utilizing AC-coupled serial channels for efficient multi-Gb/s performance. By placing coupling capacitors between the serial transmitter and receiver, AC-coupled designs remove the stringent common-mode specifications typical of DC-coupled channels [37, 38], improving system efficiency and interoperability while isolating the receiver for optimal biasing [39]. In addition, capacitive serial links are also leveraged in close-proximity chip-to-chip [40] and chip-to-board [41] communications by utilizing the capacitance between narrowly spaced pads to realize high density I/O.

In such systems, a constant design challenge is the area-efficient integration of coupling capacitors. As depicted in Fig. 1.3(a), traditional AC-coupled data links rely on large, board mounted capacitors ($C_C \geq 1\ nF$) and DC balanced encoding to set the channel cut-off frequency below that of the data stream [42]. In this way, the exponential decay of the coupled data is negligible, avoiding the problematic effects of baseline wander. However, this design methodology consumes large amounts of board area, introduces reflections into high-speed channels [43], and is impractical for high density or 3-D integrated designs. Instead, recent topologies have sought to integrate the coupling capacitor within the receiver by significantly reducing its size. To avoid baseline wander in fully integrated designs, two distinct approaches have been taken: those centered around 1) DC compensation and large RC time-constants or 2) pulse reception with a much smaller RC product.

Described in [44], the large time-constant approach offsets the reduced coupling capacitance with an increased resistance ($\tau = RC_C$), maintaining the channel cut-off frequency well below that of the encoded data stream. Supplementing this approach,
feed-back or feed-forward DC compensation (Fig. 1.3(b)) further reduces baseline variations [45]. This is demonstrated in [46] where a 5-tap low-pass feedback network actively restores the attenuated DC content by applying an analog correction voltage to the coupled input. While effective, the associated processing lowers power efficiency and requires adjustment with changing signal amplitude. Instead, [39] presents a passive feed-forward network to restore the signal DC without significant power overhead or adjustment. However, both of these feed-forward and feed-back approaches require complex adaptive circuitry at the receiver input. Additionally, the large time-constant
approach still utilizes data encoding which further increases receiver complexity and lowers the effective data rate of the overall system.

As an alternative, pulse receivers minimize the time-constant of the coupled input such that each bit fully decays within one unit interval (UI). In this way, the integration of the channel coupling is enabled by capacitances in the pF to fF range. Additionally, the resulting return-to-zero (RZ) waveform removes the effects of baseline wander. To ensure accurate data reception, a latch or 1-tap decision feedback equalizer (DFE) is used to translate the three-level coupled pulse into a binary (NRZ) output, as depicted in Fig. 1.3(c). In contrast to DC compensation, pulse receiver topologies are simple, low power, and, because of their internal latching, do not require data encoding [47].

These advantages are highlighted in [48] and [49] where respective 1.3 Gb/s and 1.4 Gb/s face-to-face data links use inverter feedback as low-overhead DFE circuits. The pulse receiver in [50] also utilizes an inverter gain stage along with a cross-coupled PMOS load, forming an asynchronous latch that operates at 3 Gb/s. Though effective, these inverter topologies have high parasitic capacitance at the receiver input, limiting receiver bandwidth and sensitivity. The work in [51] overcomes this limitation by implementing a sense amplifier and latch after the inverter stages, unloading the receiver input and achieving 10 Gb/s operation. However, the sensitivity and duty-cycle of this digital architecture are dependent on process, voltage, and temperature (PVT) variation while the single ended input is susceptible to common-mode (power and ground) fluctuations.

The pulse receiver in [52] utilizes a fully differential topology with a cross-coupled NMOS latch, rejecting common-mode noise and achieving rates up to 6 Gb/s. In [53,
two differential amplifiers are implemented in parallel, latching incoming pulses with programmable hysteresis. The parallel amplifiers utilize a high-gain, “split-load” design to further reduce loading and increase bandwidth, realizing full and half-rate speeds of 10 $Gb/s$ and 16.67 $Gb/s$, respectively. Despite data rates > 10 $Gb/s$, the power consumption of the multi-stage parallel architecture limits efficiency to $2 – 3 mW/Gb/s$.

1.3.2 The Proposed AC-Coupled Receiver

In the proposed work, a fully-integrated, differential pulse receiver is implemented, employing a single-stage, high-gain sense amplifier to maximize power efficiency. To reduce input loading, low capacitance bias switches are implemented, creating a DFE within the amplifier and removing the need for a separate output latch. Unlike previous work, the proposed design also utilizes replica biased common-mode feedback (CMFB) to ensure performance across PVT variation. With this operational stability, a low and high power mode are implemented by scaling the supply voltage, allowing the design to run at high speed or high efficiency ($< 1 mW/Gb/s$). The receiver circuit is designed in a 0.13 $\mu m$ SiGe process and utilizes the available HBTs for their high switching speeds and low parasitic capacitance. Integrated AC coupling uses high-density MIM capacitors sized to ensure pulse decay, thus minimizing baseline wander and data dependent jitter. The design is mounted on a PCB test-bed and is demonstrated at data rates from 1 – 10 $Gb/s$ and achieves a BER $< 10^{-13}$ and a peak efficiency of 0.54 $mW/Gb/s$. 

14
1.4 Outline

The scope of this research includes the conceptual design, circuit implementation, and testing of the proposed pulse receiver, interleaved $\Delta\Sigma$ modulation, and 3-bit RF-DAC, as illustrated by the overview in Fig. 1.4. Chapter 2 focuses on the early design and validation of the pulse receiver while Chapters 3 discusses the theoretical underpinnings of the interleaved DSM and DAC. Chapters 4 and 5 detail the circuit design and testing of the first and second DAC prototypes, respectively, comparing their performance with other state-of-the-art designs. Finally, Chapter 6 gives a summary of the work along with future research goals for the proposed multi-mode, interleaved $\Delta\Sigma$ DAC.

Figure 1.4: The scope of work: the interleaved multi-mode $\Delta\Sigma$ RF-DAC and the associated high-speed I/O.
Chapter 2: AC Coupled Data Input

The AC coupling of high speed I/O channels facilitates receiver design by removing common-mode range requirements and allowing the receiver bias to be optimized for high-speed or low-power operation. However, the addition of coupling capacitors creates a high-pass channel response which can cause baseline wander when fed with unbalanced NRZ data or with long strings of consecutively identical digits (CIDs). Typically, schemes such as 8b/10b encoding are used to ensure DC balance and to limit the low frequency content of the data stream. Large capacitors can then be used to couple in the data, minimizing voltage droop.

The need for this design methodology is highlighted in Fig. 2.1 where (a) an AC coupled receiver is fed with (b) a string of CIDs of amplitude $V_i$ causing (c) the coupled common-mode to deviate from the desired input bias $V_B$. Using the $RC$ input model given in Fig. 2.1(a), the highpass transfer function of an AC coupled channel can be approximated by (2.1). In this equation, the coupling capacitance is represented by $C_C$, the termination resistance by $R_T$, and the parasitic input capacitance by $C_P$.

$$\frac{V_{INC}}{V_{IN}} = \frac{sR_TC_C}{sR_T(C_C + C_P) + 1} \quad (2.1)$$

By modeling an incoming bit as a step function, the exponential decay of the coupled input is expressed in (2.2), having the time constant $\tau$ given in (2.3). The typically
Figure 2.1: (a) The RC input model of an AC coupled receiver. (b) Receiver input containing a string of CIDs of amplitude $V_i$ and (c) the coupled input showing exponential decay and baseline wander.

A large $C_C$ dominates the input response, allowing the coupled signal to be designed such that the voltage droop is below some maximum tolerance [55].

$$V_{INC}(t) = \frac{V_i}{2} \left( 1 - e^{\frac{-t}{\tau}} \right)$$

(2.2)

$$\tau = R_T (C_C + C_P)$$

(2.3)


2.1 Pulse Receiver

Unlike traditional AC coupling, however, the goal of a pulse receiver is not minimum voltage droop (via a large coupling capacitance), but instead a DC balanced RZ waveform, eliminating the common-mode variations caused by CIDs. To create this RZ pulse, a small $C_C$ is used to ensure incoming bits are coupled into the receiver but fully decay within one bit period ($t_B$). Such a design is ideal for fully-integrated chip-to-chip I/O, as it reduces capacitor size, but a thorough modeling of the input is needed to ensure sufficient coupling with minimal jitter. To this end, a rise time $t_R$ is added to the model of the incoming bit, creating a more accurate representation of high-speed data. As shown in Fig. 2.2(a), this waveform is applied to the $RC$ input model, the slope ($m$) of the transition set by the input amplitude and rise time ($m = V_i/t_R$). The coupled signal, given by (2.4), is illustrated in Fig. 2.2(b) and is comprised of two parts: the transition response ($t < t_R$) and the subsequent decay ($t_R < t < t_B$).

\[ V_{IN} \]

\[ m \]

\[ V_i \]

\[ t_R \]

\[ t_B \]

\[ V_{INC} \]

\[ V_C \]

\[ V_E \]

(a) (b)

Figure 2.2: (a) The input waveform representing high-speed data with a finite rise time and (b) the partially coupled input.
\[ V_{INC}(t) = mR_TC_C \left( 1 - e^{-\frac{t}{\tau}} \right) - u(t - t_R) \cdot mR_TC_C \left( 1 - e^{-\frac{(t-t_R)}{\tau}} \right) \] (2.4)

During the transition response, an incoming bit is only partially coupled into the receiver due to the small size of \( C_C \). The peak coupled amplitude \( V_C \) can be found by evaluating (2.4) at time \( t_R \), resulting in the expression given in (2.5). This peak amplitude, along with the gain of the receiver, directly sets the sensitivity of the design with increasing values of \( C_C \) contributing to a more sensitive receiver.

\[ V_C = \frac{V_i R_TC_C}{t_R} \left( 1 - e^{-\frac{t_B}{\tau}} \right) \] (2.5)

After the transition response, the coupled signal decays toward its bias point where, at the end of the bit period, some residual error voltage \( V_E \) remains. This error represents a departure from the intended RZ signaling and causes energy from one bit to leak into the next, resulting in ISI jitter. The magnitude of \( V_E \) can be determined by solving (2.4) at \( t_B \), shown in (2.6).

\[ V_E = \frac{V_i R_TC_C}{t_R} \left( e^{\frac{t_B-t}{\tau}} - e^{-\frac{t_B}{\tau}} \right) \] (2.6)

As with \( V_C \), a higher coupling capacitance increases \( V_E \), leading to a direct trade-off between receiver sensitivity and ISI jitter in the sizing of the input capacitor. This is demonstrated in Fig. 2.3(a-b) where the coupled amplitude and the error voltage are plotted for different values of \( C_C \) across bit rate. For these figures, \( C_P \) is set at 10% of \( C_C \) and \( t_R \) is kept at 20% of \( t_B \). As \( C_C \) is increased, sensitivity is improved at the expense of jitter. This design trade-off is further complicated by increases in data rate where a shrinking \( t_B \) reduces settling time, increasing \( V_E \) further.
To find the deterministic jitter added by a given $V_E$, the difference in crossing time ($\Delta t$) between an ideal transition and one with $V_E$ is calculated. Depicted in Fig. 2.3(c), this timing variation can be described in (2.7) by equating these two transition cases. Solving for $\Delta t$ (i.e. the peak-to-peak jitter) results in the expression given in (2.8). Figure 2.3(d) plots this jitter, indicating that a 1.5 pF coupling capacitor will experience only 20% peak-to-peak jitter at 10 Gb/s with minimal impact to the coupled amplitude.
\[ V_{INC}(t) = V_{INC}(t + \Delta t) - V_E \] \quad (2.7)

\[ \Delta t = -\tau \ln \left( \frac{-V_E t_R}{V_i R_T C_C} + e^{-\frac{t}{\tau}} \right) - t \] \quad (2.8)

### 2.1.1 Latching

As detailed in Chapter 1.3, many pulse receivers utilize latches after the receiver amplifier to hold the incoming pulses for an NRZ output. These single-ended, predominantly digital architectures suffer from PVT variates and common mode disturbances. To overcome this, \cite{53, 54} implement multistage, parallel amplifiers to latch data within the gain stage. However, the multistage topology only achieves an efficiency of 2-3 mW/Gb/s. The proposed architecture implements a bias latch fully integrated into the single stage receiver amplifier, depicted in Fig. 2.4, realizing a robust, low power design. Two separate bias voltages \( V_{B1} \) and \( V_{B2} \) are selectively applied to the differential inputs of the receiver to latch previous bit values. In this way, the positive and negative input of the receiver do not decay completely to the same potential with the difference between the two biases \( \Delta V_B \) setting the receiver hysteresis. To select which bias is applied to each input, the receiver output is fed back to the bias switches. The effect of this toggled bias is a latching receiver which accurately handles strings of CIDs without encoding.
2.2 The Bias Latched Pulse Receiver

A functional diagram of the proposed pulse receiver is shown in Fig. 2.5. Following the on-chip AC coupling, a high-gain differential amplifier recovers the incoming low-voltage pulses. A level-shifting buffer then translates the signal to CMOS levels where cascaded inverters create the full-scale digital output. Received data is fed back from these inverters to switches within the amplifier, toggling its bias point to latch the data. Common mode feedback (CMFB) is employed to lock the buffer common mode to the switching threshold of the inverters. This guarantees constant duty-cycle across process, voltage, and temperature (PVT) variation and allows the receiver to operate as the supply voltages are adjusted between low power and high power operation.
**Table 2.1: Receiver Circuit Sizing**

<table>
<thead>
<tr>
<th>Devices</th>
<th>Size</th>
<th>Components</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Q_1$, $Q_2$</td>
<td>$L: 3 \mu m$ $W: 120 nm$</td>
<td>$R_1$, $R_2$</td>
<td>800 $\Omega$</td>
</tr>
<tr>
<td>$Q_3$, $Q_4$</td>
<td>$L: 5 \mu m$ $W: 120 nm$</td>
<td>$R_3$, $R_4$</td>
<td>145 $\Omega$</td>
</tr>
<tr>
<td>$M_1$, $M_3$</td>
<td>$W: 2 \mu m$ $L: 240 nm$</td>
<td>$R_5$, $R_6$</td>
<td>60 $\Omega$</td>
</tr>
<tr>
<td>$M_2$, $M_4$</td>
<td>$W: 6 \mu m$ $L: 240 nm$</td>
<td>$R_7$, $R_8$</td>
<td>1 $k\Omega$</td>
</tr>
<tr>
<td>$M_5$</td>
<td>$W: 18 \mu m$ $L: 150 nm$</td>
<td>$R_9$, $R_{10}$</td>
<td>25 $k\Omega$</td>
</tr>
<tr>
<td>$M_6$, $M_8$, $M_{10}$</td>
<td>$W: 5 \mu m$ $L: 120 nm$</td>
<td>$R_{11}$</td>
<td>2.3 $k\Omega$</td>
</tr>
<tr>
<td>$M_7$, $M_9$, $M_{11}$</td>
<td>$W: 1.7 \mu m$ $L: 120 nm$</td>
<td>$C_C$</td>
<td>1.4 $pF$</td>
</tr>
</tbody>
</table>

The full receiver schematic is shown in Fig. 2.6 and Table 2.1 gives the device sizes and component values used.

### 2.2.1 Input Amplifier

The core of the design is the input amplifier which is created from a single differential pair. The amplifier utilizes SiGe HBTs ($Q_1 - Q_2$) for their superior switching speed and low parasitic capacitances, extending the amplifier bandwidth beyond that of the available CMOS and increasing the maximum data rate of the design.
To bias the amplifier, the proposed topology creates an offset ($\Delta V_B$) between the positive and negative amplifier bias points, forcing opposite polarity inputs to decay to different steady-state voltages. Figure 2.7 shows a simulation of the coupled input ($IN_{C+}$) where the effects of the offset are highlighted by long runs of both 0’s and 1’s. The input pulses decay, but a minimum difference is maintained and subsequently amplified by the input stage. For compatibility with the hysteresis requirements of LVDS [37] and other standards, the magnitude of this offset is set to $\sim 25 \text{ mV}$, thus removing data reception errors caused by noise and line reflections.

The two receiver biases $V_{B1}$ and $V_{B2}$ are created on chip via ratioed resistors. With this method of implementation, the biases can track supply variations as the receiver voltages are tuned for low or high power operation. Transistors $M_1 - M_4$ are used as the bias switches and are composed of thick gate PMOS devices to guard against breakdown. To reduce the capacitive loading on the digital feedback, $M_1 - M_4$ are implemented as small, high resistance switches with their $R_{ON}$ integrated into the
bias generation network. The switches are interdigitated to minimize unwanted bias offsets caused by mismatch.

2.2.2 Level Shifting and CMFB

Following the input gain stage, two common collector buffers level-shift the data down to CMOS levels. Of critical importance is the centering of the output common mode on the inverter mid-rail voltage. Due to variations in PVT, the common mode can drift away from this mid-rail, closing the receiver eye with duty cycle distortion, as shown in Fig. 2.8. A CMFB amplifier, labeled in Fig. 2.6, is used to sense the buffer common mode and compare it to a self-biased replica of the first inverter stage. By adjusting the current, and thus the output level, of the input amplifier, the buffer output is pinned to the mid-rail of the first inverter. Since the CMFB only compensates for low-frequency variation, the bandwidth of the feedback loop is kept below 50 MHz so that it does not interfere with the high speed bias control.
Additionally, the gain and phase margin of the CMFB are set to $28 \, dB$ and $\sim 60^\circ$, respectively, ensuring stable operation.

After the common collector buffer, a chain of inverters conditions the data and drives the PMOS bias switches. The inverters are optimized for minimum feedback latency, achieving a delay of $\sim 20 \, ps$ each. The effect of this feedback delay can be seen in the input simulation of Fig. 2.7 where the incoming pulse droops below the desired bias before the switches are toggled. While this voltage droop contributes to the receiver output jitter, it is kept below $5 \, mV$ by an optimized feedback path, ensuring operation up to $10 \, Gb/s$.

### 2.2.3 Simulation

The receiver performance is verified through extracted simulation where the receiver is driven by a random bit generator via modeled wirebonds and transmission lines, the output taken after the cascaded inverter chain. Figure 2.9 gives the receiver output eye diagram after $10^4$ random bits for both low power ($V_{CC} = 1.7 \, V$) and
Figure 2.9: Extracted simulation of the receiver eye diagram at $V_{CC} = 1.7\,V$ and $2.1\,V$, $7.5\,Gb/s$ and $10\,Gb/s$ respectively.

high power ($V_{CC} = 2.1\,V$) operation. In the low voltage mode, data dependent jitter caused by the bias feedback latency is evident in the output eye and results in a peak-to-peak jitter of $15\,ps$. While this deterministic jitter decreases the eye opening, it does not compromise receiver functionality. The high voltage mode exhibits this behavior as well, but the increased data rate and random jitter of this operating point obscures any data dependency and results in a simulated jitter of $18\,ps$, very close to the estimated jitter of $20\,ps$ calculated in Fig. 2.3.
2.3 LVDS Receiver Testing

The proposed LVDS pulse receiver is implemented in a 0.13 $\mu m$ SiGe BiCMOS process where it occupies 0.0115 $mm^2$. Dominated by the area of the MIM coupling capacitors (0.00952 $mm^2$), the receiver layout is placed between the two differential input pads for high density integration, as shown in Fig. 2.10. To measure the receiver performance, the captured data is looped back off-chip without clock alignment. This is done so that the system jitter can be measured off-chip and a BER test can be performed. A CML differential pair is used as a 50 $\Omega$ buffer, retransmitting the full-scale receiver output. While this buffer is only 25% efficiency [56], it provides a simple, robust means of data transmission. From simulation, the jitter added by the buffer is negligible compared with that of the receiver, thus the measured system jitter is assumed to be predominantly a result of the receiver.

The test chip containing the receiver and loop-back buffer is wire-bonded to a two layer FR4 printed circuit board (PCB). Shown in Fig. 2.11, the first PCB layer provides power, ground, and $2.5 cm$ 50 $\Omega$ transmission lines to SMA connectors. The second layer is used only as a transmission line ground for the high-speed data lines, ensuring matched line impedances. Matched 1 $m$ cables are then used to connect the design to a digital signal analyzer or BER tester for measurement.
Figure 2.10: The receiver test chip.

Figure 2.11: The receiver test board.
2.3.1 Measurement Results

The loop-back testing depicted in Fig. 2.12 is accomplished with an input pseudo random bit sequence (PRBS) of $2^{23} - 1$ bits, used to exercise the receiver with long runs of invariant data. The design is tested under the two supply conditions outlined in Table 2.2. In the low power mode, an analog voltage of 1.7 V and a digital voltage of 1.1 V are used, resulting in a total power consumption of 4 mW. Under this condition, the receiver achieves a maximum data rate of 7.5 Gb/s with a BER of $< 10^{-13}$ and an efficiency of 0.53 mW/Gb/s. Figure 2.13 shows the output eye of the receiver/output buffer system at 3 Gb/s and at 7.5 Gb/s. At these data rates, a peak-to-peak jitter of 31.1 ps and 45.6 ps is recorded.
Table 2.2: Receiver Low and High Power Settings

<table>
<thead>
<tr>
<th>Receiver Settings</th>
<th>Low Power</th>
<th>High Power</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analog Voltage ($V_{CC}$)</td>
<td>1.7 V</td>
<td>2.1 V</td>
</tr>
<tr>
<td>Digital Voltage ($V_{DD}$)</td>
<td>1.1 V</td>
<td>1.5 V</td>
</tr>
<tr>
<td>Power</td>
<td>4 mW</td>
<td>9.5 mW</td>
</tr>
<tr>
<td>Data Rate</td>
<td>7.5 Gb/s</td>
<td>10 Gb/s</td>
</tr>
</tbody>
</table>

At bit rates above 7.5 Gb/s, the receiver eye diagram begins to degrade and the BER increases as a result. From simulation, the limiting factor in the low power mode is the drive strength of the receiver output inverters. With only a 1.1 V supply, these inverter buffers cannot drive the input capacitance of the CML transmitter at speed. This issue is solved in the high-power mode by increasing both the analog and digital supply voltage to 2.1 V and 1.5 V, respectively.

In the high power mode listed in Table 2.2, the maximum bit rate increases to 10 Gb/s while the receiver power consumption increases to 9.5 mW. Again, a BER of $< 10^{-13}$ is recorded while an efficiency of 0.95 mW/Gb/s is achieved. Figure 2.14 gives the measured eye diagrams of the system at 7 Gb/s and at 10 Gb/s. With the higher supply voltages, the peak-to-peak jitter at these two data rates is 29.4 ps and 52.8 ps, respectively.

In Fig. 2.15, the peak-to-peak jitter measured during loop-back testing is plotted as a percentage of the UI from 1 – 10 Gb/s (plots from Fig. 2.13 and Fig. 2.14 are labeled). At low data rates, the low power jitter is less than that of the high power, resulting in high-speed, power efficient data reception. The higher jitter present in the high-power mode is due to its increased $\Delta V_B$ from the raised supply voltage. However, as data rates increase above 7 Gb/s, the low power jitter exceeds that of the high power mode and the increased supply voltage is needed to maintain operation.
Figure 2.13: Low power output eye of the receiver at (a) 3\,Gb/s and at (b) 7.5\,Gb/s.

Figure 2.14: High power output eye of the receiver/transmitter at (a) 7\,Gb/s and at (b) 10\,Gb/s.

Figure 2.15: Peak-to-peak jitter measurements from 1 to 10\,Gb/s in low and high power mode.
2.3.2 Comparison

Table 2.3 compares the proposed architecture to other Gb/s pulse receivers. With the exception of [51] which does not include the receivers on-chip buffer, the design achieves the highest power and area efficiency. Additionally, the proposed receiver is implemented with a single stage differential amplifier with CMFB, minimizing mismatch errors and achieving an input data rate of 10 Gb/s. With this robust input cell, the proposed ∆Σ DAC can be fed with real-time high-speed data, avoiding the area consumption of highly parallel I/O or on-chip memory while also minimizing design complexity by removing data encoding.

Table 2.3: Comparison of Gb/s Pulse Receivers

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>[41]</td>
<td>350 CMOS</td>
<td>2.5 Gb/s</td>
<td>&lt;10^{-12}</td>
<td>&lt;120</td>
<td>10.3</td>
<td>4.1</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>[47]</td>
<td>100 CMOS</td>
<td>1 Gb/s</td>
<td>-</td>
<td>-</td>
<td>2.7</td>
<td>2.7</td>
<td>0.0402</td>
<td>0.0281</td>
<td>28,050</td>
</tr>
<tr>
<td>[48]</td>
<td>350 CMOS</td>
<td>1.35 Gb/s</td>
<td>&lt;10^{-12}</td>
<td>-</td>
<td>3.6</td>
<td>2.7</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>[50]</td>
<td>180 CMOS</td>
<td>3 Gb/s</td>
<td>&lt;10^{-12}</td>
<td>&lt;120</td>
<td>10</td>
<td>3.33</td>
<td>0.0043</td>
<td>0.0007</td>
<td>225</td>
</tr>
<tr>
<td>[53]</td>
<td>90 CMOS</td>
<td>10 Gb/s</td>
<td>-</td>
<td>35*</td>
<td>32</td>
<td>3.2</td>
<td>0.045</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>[53]</td>
<td>180 CMOS</td>
<td>3.3 Gb/s</td>
<td>-</td>
<td>55.6</td>
<td>40</td>
<td>12.1</td>
<td>0.97†</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>[54]</td>
<td>90 CMOS</td>
<td>13 Gb/s</td>
<td>-</td>
<td>35*</td>
<td>26</td>
<td>2.0</td>
<td>0.6</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>[51]</td>
<td>90 CMOS</td>
<td>10 Gb/s</td>
<td>&lt;10^{-14}</td>
<td>45*</td>
<td>2.7*</td>
<td>0.27†</td>
<td>-</td>
<td>0.0012</td>
<td>120</td>
</tr>
</tbody>
</table>

This Work

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>(LP)</td>
<td>SiGe</td>
<td>7.5</td>
<td>&lt;10^{-13}</td>
<td>45.6</td>
<td>4.0</td>
<td>0.54</td>
<td>0.0115</td>
<td>0.002</td>
<td>261</td>
</tr>
<tr>
<td>(HP)</td>
<td>BiCMOS</td>
<td>10</td>
<td>&lt;10^{-13}</td>
<td>52.8</td>
<td>9.5</td>
<td>0.95</td>
<td>196</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

* Estimated From Figure  † Buffer Not Included  ‡ Includes Pads
Chapter 3: The Interleaved, Multi-mode \(\Delta \Sigma\) DAC

In the pursuit of increasingly digital transmitter architectures, both \(\Delta \Sigma\) modulation [57] and interleaving [58] have been explored as a means of improving the speed and in-band performance of DACs through highly-scalable, digital topologies. Previously, these two concepts have been handled separately or together as a means of increasing \(\Delta \Sigma\) modulator (DSM) performance for baseband applications [29]. In this work, these two techniques are for the first time independently combined in pursuit of a reconfigurable method of direct digital-to-RF synthesis. As stated in Chapter 1.2.4, the confluence of these two techniques provides a unique method of low-noise signal generation which is suitable for a wide range of wireless applications. In this chapter, the proposed DAC design is discussed in conceptual terms with the first section highlighting the reconfigurable error feedback DSM utilized by the design and the second highlighting the ideal and non-ideal interleaving of independent \(\Delta \Sigma\) modulated channels. Following this, a brief comparison of the proposed design and a typical Nyquist rate architecture is given.

3.1 Reconfigurable \(\Delta \Sigma\) Modulation

For \(\Delta \Sigma\) modulation to be a viable part of a direct digital-to-RF system, the DSM must support both GHz sample rates and frequency-agile passband synthesis.
From the many ΔΣ topologies available, the error feedback DSM shown in Fig. 3.1 is used for this application as its simple architecture lends itself to high-speeds and re-programmable outputs (i.e. short critical paths in feedback and straight-forward digital implementation) [59]. Its structure is composed of three independent operations: quantization, feedback filtering, and input summation; all of which can be implemented as reconfigurable digital blocks that are optimized for high-speed clocking.

As an overview of the error feedback structure in Fig. 3.1, the modulator output $y_1[n]$ is composed of $N$ bits and is derived from two internal relationships: the error generation given in (3.1) and the input summation in (3.2).

\[
\varepsilon_1[n] = y_1[n] - u_1[n] \quad (3.1)
\]

\[
u_1[n] = x_1[n] - \varepsilon_1[n] \cdot G(z) \quad (3.2)
\]
In these expressions, a uniform linear quantizer is assumed with the $T$-bit quantization error represented by $\varepsilon_1[n]$, the $M$-bit modulator input by $x_1[n]$, and the $N+T$ bit post-summation signal by $u_1[n]$. By substituting (3.1) into (3.2), a relationship between the modulator input and output is obtained in (3.3).

$$y_1[n] = x_1[n] + \varepsilon_1[n] (1 - G(z))$$

(3.3)

From this expression, it can be seen that $x_1[n]$ appears at the modulator output without distortion or delay\(^1\) while the noise transfer function ($NTF = 1 - G(z)$) shapes the frequency content of $\varepsilon_1[n]$, creating a low-noise bandwidth. With proper design, this shaping function can be used to create a low-noise RF passband in the error spectrum. To achieve reconfigurable operation of this passband, several shaping functions are implemented concurrently, differing by only delays ($z^{-1}$) and polarity ($\pm$). Selecting among these related functions allows a single DSM to realize a variety of RF passbands with minimal overhead.

This section describes the operation of the error feedback DSM by highlighting the three components of its architecture. Additionally, the reconfigurable noise shaping achieved by the proposed design is detailed.

### 3.1.1 Digital Quantizer

At the output of the error feedback DSM is the quantization and error generation block illustrated in Fig. 3.1. Since both the input and output of the quantizer are digital, the quantization process can be viewed as a second quantization of an already discrete signal. This accurately describes the loss of resolution which occurs between $x_1[n]$ experiences no latency in (3.3).
Figure 3.2: (a) A digital quantizer functioning over the range $[0, 2^N)$ with a bin width of $\Delta_N = 1$. The error values (the $T$ LSBs) are considered fractional. (b) Digital truncation, (c) the truncation transfer function, and (d) the probability mass function (PMF) of $\varepsilon_1[n]$ assuming a “busy” $u_1[n]$.

The $N+T$ bit input and the $N$ bit output, as shown in Fig. 3.2(a). Because of the high sample rates needed for direct RF synthesis, quantization and error generation are most efficiently accomplished by truncating the $T$ least significant bits (LSBs) of $u_1[n]$. Shown in Fig. 3.2(b), the discarded LSBs are subsequently used as the error signal, greatly simplifying the error generation process [59].

The transfer function describing the truncation quantizer is given in Fig. 3.2(c), having a valid input range$^2$ of $[0, 2^N)$. The values of $\varepsilon[n]$ that result from truncation

$^2$The $N$ MSBs are considered whole number while the $T$ LSBs are considered fractional ($N.T$).
are of the form \(-i/2^T\) where \(i\) is an integer such that \(i \in [0, 2^T-1]\). Assuming a modulated or “busy” input (i.e. at any time its position within the quantizer interval \(\Delta_N\) is essentially random) with \(T\) sufficiently large, the error can be considered stochastic [60], exhibiting the probability mass function shown in Fig. 3.2(d). The error signal can then be modeled as an additive white noise introduced by the truncation quantizer (as done in (3.1)). The power of this added noise is given in (3.4) by calculating the mean-squared value of Fig. 3.2(d) and removing its DC offset.

\[
\varepsilon_{RMS}^2 = \frac{1}{2^T} \cdot \sum_{i=0}^{2^T-1} \left(\frac{-i}{2^T}\right)^2 - \left(\frac{2^T-1}{2^T+1}\right)^2 = \frac{\Delta_N^2}{12} - \frac{\Delta_{N+T}^2}{12} \tag{3.4}
\]

Since the truncation quantizer operates on a discrete signal, the expression in (3.4) represents only the noise power introduced by truncation and not the total noise power in the output \(y[n]\). This introduced noise can be viewed as the difference between the output noise (\(\Delta_N^2/12\)) and the quantization noise initially present in the discrete input (\(\Delta_{N+T}^2/12\)), where \(\Delta_{N+T} = \Delta_N/2^T\). This distinction is of critical importance given that the digital DSM can only shape the introduced noise (\(\varepsilon_{RMS}^2\)) and not the total noise power. For this reason, the \(M\)-bit input resolution of the DSM sets the minimum attainable noise floor by determining the amount of unshaped noise in the input \(x[n]\).

### 3.1.2 Feedback Filter

After truncation, the quantization error is passed through a feedback filter, shaping it away from the desired passband. Because of its high rate of operation, the feedback filter requires a simple design which lends itself to reconfigurable passband synthesis. The lowpass (LP) filter function shown in (3.5) satisfies both of these
requirements. By creating an $\alpha$ order filter with only whole-number, binomial coefficients, a digital implementation of (3.5) requires only adds and shifts which facilitates a programmable high-speed design.

$$G(z)_{LP} = 1 - (1 - z^{-1})^\alpha$$  \hspace{1cm} (3.5)

Fig. 3.3 shows the realization of this feedback filter for both first and second order designs while Fig. 3.4(a) depicts the noise shaping and pole/zero location that characterize them. The shaped PSD in Fig. 3.4(a) is plotted as a function of the system sample frequency $f_S$ (using the identity $z = e^{j2\pi f_S}$) across the first two Nyquist zones and is normalized by the unshaped power of $\varepsilon_1[n]$. In the first Nyquist zone, the output spectrum exhibits a low-noise passband of width $f_B$ near DC while the gain of the filter amplifies the out-of-band noise, reaching its maximum at $f_S/2$.

Due to the sampled nature of the system, the frequency content of the first Nyquist zone is mirrored about $f_S/2$, producing additional passbands every multiple of $f_S$. While the first passband (labeled in Fig. 3.4(a)) is of little use for RF synthesis, higher passbands easily reach into the RF domain for $f_S$ at GHz frequencies. These higher Nyquist outputs hint at the DSMs ability to generate RF signals from image
Figure 3.4: (a) Shaping of a first (dashed line) and second (solid line) order LP DSM with pole/zero locations shown, \( f_B \) representing the usable ∆Σ bandwidth. (b) BP and HP DSM shaping with associated pole/zero locations.

replicas [26, 58]. Despite this potential for RF outputs, the filter function given in (3.5) exhibits sparse spectrum coverage with large portions of the Nyquist band unfit for RF synthesis due to high out-of-band noise. A DSM using only (3.5) as the feedback filter would need to widely vary \( f_S \) to achieve a reconfigurable RF output. In addition to the inherent difficulties with generating a low-noise, widely-variable system clock, this method of agile RF synthesis is problematic due to its large impact on both the system sample rate and output bandwidth.
Instead, this work proposes the use of multiple filter functions to improve the DSM’s coverage of the output spectrum and minimize the tuning of $f_S$. Using the substitutions $z^{-1} \rightarrow -z^{-2}$ and $z^{-1} \rightarrow -z^{-1}$ in (3.6) and (3.7), bandpass (BP) and highpass (HP) filter functions can be synthesized, respectively, from the original LP design. The shaping that results from these transforms is plotted in Fig. 3.4(b) and depicts the filter passbands shifted to $f_S/4$ and $f_S/2$ in the first Nyquist zone. These changes correspond to a rotation in the zero locations of the NTF, also shown in Fig. 3.4(b).

\[
G(z)_{BP} = 1 - (1 + z^{-2})^\alpha
\]  

(3.6)

\[
G(z)_{HP} = 1 - (1 + z^{-1})^\alpha
\]  

(3.7)

Figure 3.5: Reconfigurable feedback filter implementing LP, BP, and HP functions to synthesize a variety of low-noise passbands with $\alpha = 2$. 

\[\text{Normalized Error Power}\]
\[\text{Normalized Frequency}\]
Taken as a whole, these three filters synthesize output bands at every multiple of \( fs/4 \). This is illustrated in Fig. 3.5 where the feedback filter selects among the three shaping functions to realize one of several RF passbands. Of note in the output of the reconfigurable DSM are the contiguous passbands of the LP and HP filters. For these two filter configurations, multi-Nyquist operation doubles their bandwidth as they are now composed of passbands in adjacent Nyquist zones. While filter functions capable of synthesizing other passbands (e.g. \( 5fs/8 \)) are possible in this configuration [61], they do not lend themselves to high-speed implementation due to their coefficients and thus are not considered here.

Because of the similarity between the LP, BP, and HP filter functions, a reconfigurable implementation is straight-forward. Figure 3.6 illustrates a reconfigurable design of both the first and second order DSM previously shown in Fig. 3.3 (second and fourth order in the BP case).

Figure 3.6: Block diagrams of (a) first order and (b) second order reconfigurable error feedback DSMs.
### 3.1.3 Input Summation

After the shaping of the feedback filter, the quantization error is summed with the modulator input, closing the feedback loop. Since this summation feeds directly into the quantizer, as shown in Fig. 3.1, the resulting sum must lie completely within the range of the quantizer. This is of critical importance if the quantizer is realized by truncation because overflow or underflow does not just saturate the quantizer; it results in wrap-around and leads to inaccurate modulator outputs. To prevent this, an upper limit is placed on the value of $x[n]$ such that overflow becomes impossible under any feedback condition.

Equation (3.8) gives this maximum input value, derived by rearranging the input summation in (3.2) and substituting the maximum feedback value and quantizer range. As before, the truncated LSBs are assumed fractional.

\[
x_{MAX} = \frac{(2^{N+T} - 1) - (2^\alpha - 1)(2^T - 1)}{2^T}
\]

From (3.8) it can be shown that $x_{MAX}$ will have a maximum bus width of $N + T$ bits, leading to the equivalence $M = N + T$. For negative values of $x_{MAX}$, no amount of backoff ensures proper quantizer operation. Because of this, the number of output bits must be greater than or equal to the filter order ($N \geq \alpha$), resulting in a positive maximum input. This can be seen in Fig. 3.7 where (3.8) is plotted as a percent of the fullscale (FS) input. As $\alpha$ is increased, the amount of back-off needed also increases, significantly reducing signal power for higher orders of noise shaping.

While this analysis assumes the inputs to the summation are positive, it is also valid for negative inputs which result from the BP or HP filter coefficients. By ensuring $x[n]$ has some static offset value to guarantee a positive $u[n]$, $x_{MAX}$ can be
Figure 3.7: Maximum input, as a percent of FS, under which quantizer overflow is eliminated. This is plotted across varying $N$ and $\alpha$, assuming $T = 12$.

viewed as a valid input range and the conditions described in (3.8) remain valid. In this way, the input summation (or subtraction) is designed for all three filters and guarantees proper modulator operation.

3.2 Multi-Nyquist Operation

As previously described in Section 3.1.3, the multi-Nyquist spectrum of the reconfigurable DSM exhibits low-noise passbands at multiples of $f_S/4$. By setting $f_S$ in the GHz range, these passbands are generated at select RF frequencies and achieve low-noise outputs due to high OSRs. As an example of the operation of the reconfigurable modulator, Fig. 3.8(a) shows a BP-configured DSM generating passbands at odd multiples of $f_S/4$. Switching to an LP or HP configuration in Fig. 3.8(b) achieves the same noise floor as the BP DSM but at double the bandwidth, placing passbands at even or odd multiples of $f_S/2$, respectively. As the DSM sample-rate increases,
the output frequency and OSR of each passband increase as well, widening the RF bandwidth.

However, DAC replica images create a major obstacle to the use of multiple Nyquist passbands. For an intended output of frequency $f_0$, there exists replicas of this signal in every Nyquist band, the frequencies of which are represented by $f_I$ in (3.9).

$$f_I = kf_S \pm f_0 \quad [k = 1, 2, 3, ...] \quad (3.9)$$

To remove these images, high-order bandpass filtering is typically required, increasing size and complexity while limiting the reconfigurability of the system. In the case of the BP-configured DSM in Fig. 3.8(a), the filtering can be relaxed by increasing $f_S$, offsetting the signal and image frequencies by $f_S/2$. However, to make use of the passbands which straddle a Nyquist boundary, the image cannot be pushed out by increasing $f_S$. As depicted by the HP shaping in Fig. 3.8(b), the output passband is occupied by both the desired signal and the interfering image.

To mitigate this in-band replica, works such as [62] propose a modified DSM filter-function to introduce a guard band between the image and the signal. This solution sacrifices usable bandwidth while only slightly relaxing filter requirements. Instead, recent works in Nyquist DAC interleaving [27, 63] are leveraged in the proposed architecture to remove interfering images and reduce system filter requirements.

### 3.2.1 Interleaving

As described generally in [27] and specifically for a 2-channel system in [63], the addition of parallel data paths increases the effective sample rate of a system without
increasing the sample rate of any individual path. Along with this rate increase, it
then becomes possible to cancel all replica images up to the effective Nyquist-rate of
the system.

Unlike previous interleaved DSMs, the proposed architecture does not require error
values to be passed between parallel channels [29], keeping the modulators indepen-
dent of one another and removing inter-modulator speed and timing requirements.
Additionally, the independence of the interleaved channels allows the resulting output
to be easily summed, removing the need for high-speed multiplexing which unduly
increases the sample-rate of the down-stream DAC [29] and folds shaped out-of-band

Figure 3.8: The reconfigurable DSM: (a) depicts BP shaping with both the signal
and first image shown while (b) depicts LP (blue) and HP (red) shaping with the HP
signal and image in the same passband.
Figure 3.9: For interleaved operation, (a) alternate data samples aligned to 180° clocks are fed into (b) two parallel DSM and DAC channels. (c) Each DSM output spectrum is composed of the input signal $x[n_{e,o}]$ and the uncorrelated error $\varepsilon_{1,2}[n_{e,o}]$ shaped by the loop filter. Upon D/A conversion and summation, (d) sinc shaping resulting from the DAC zero-order-hold (ZOH) is applied to the output and the unwanted (anti-phase) images are canceled.

noise back in-band [64]. Instead, the DSM and DAC are interleaved together, maintaining a constant sample-rate throughout the digital and mixed-signal domains.

This scenario is depicted in Fig. 3.9(a-b) for the interleaved-by-two case. Here, the number of parallel data paths $P$ is set to 2, each path (or channel) consisting of an independent DSM and DAC. As highlighted by Fig. 3.9(a), data is alternately fed into each channel, the first channel taking the even indiced data $x[n_e]$ and the second taking the odd $x[n_o]$. Both channels operate at a clock rate of $f_s$ but are phase-offset
from one another by 180° (360°/P) to preserve the timing of the even and odd data samples. After passing through the modulators, the channelized data is converted to the analog/RF domain by the DACs, the outputs of which are summed together to cancel the interfering images.

Before the D/A conversion depicted in Fig. 3.9(b), the outputs of the two DSMs are represented by the discrete signals \( y_1[n_e] \) and \( y_2[n_o] \) given in (3.10) and (3.11).

\[
y_{CH1}[n_e] = x[n_e] + \epsilon_{CH1}[n_e](1 - G(z)) \quad (3.10)
\]

\[
y_{CH2}[n_o] = x[n_o] + \epsilon_{CH2}[n_o](1 - G(z)) \quad (3.11)
\]

Mirroring the output expression in (3.3), both (3.10) and (3.11) are comprised of two parts: the original inputs \( x[n_e] \) and \( x[n_o] \) and the quantization errors \( \epsilon_{CH1}[n_e] \) and \( \epsilon_{CH2}[n_o] \) shaped by their respective loop filters. For clarity, the spectrums of these signals are depicted separately in Fig. 3.9(c), their sums giving rise to the spectrums of \( y_{CH1}[n_e] \) and \( y_{CH2}[n_o] \).

In Fig. 3.9(c), the spectrums of \( x[n_e] \) and \( x[n_o] \) are composed of the desired output signal and its associated replica images. Due to interleaved sampling, the phases of these images are rotated by the sampling clock. For the 180° clock, the \( k \)th image in the \( x[n_o] \) spectrum is rotated by \( k \cdot 180° \) while the image phases in the \( x[n_e] \) spectrum are derived from the 0° clock and remain unrotated [27]. As for the noise, the spectrums of \( \epsilon_{CH1}[n_e] \) and \( \epsilon_{CH2}[n_o] \) have the same power density and shaping, as seen in Fig. 3.9(c), but are uncorrelated with one another due to the stochastic nature of the two quantization errors.
Figure 3.10: Simulation comparing a single HP DSM and DAC with an interleaved-by-two architecture. Interleaving cancels the in-band image, increasing the signal power by 6 dB and the noise floor by 3 dB.

Fed into the interleaved DACs, these DSM outputs are shaped by the hold function of the D/A conversion, which is assumed in Fig. 3.9(d) to be a zero-order-hold (ZOH). For ideal D/A conversion and summation, the desired signal and the even images add constructively, increasing their output power by 6 dB, while the odd images are antiphase and cancel. The addition of the quantization error is treated as the summation of two additive noises, raising the output noise density by 3 dB.

To demonstrate these effects, simulation results of the interleaved-by-two architecture are shown in Fig. 3.10. Using an ideal 3-bit DAC ($N = 3$) with a 1 mA $I_{LSB}$ and second order HP shaping, the simulation demonstrates complete image cancellation. In addition, the predicted increases to both the desired signal and noise power are seen in the output spectrum.

The ideal and non-ideal simulation results are obtained from the model given in Appendix A.
In general, parallel DSM and DAC channels cancel the first $P - 1$ images, clearing their respective Nyquist zones for RF synthesis. Additionally, the signal power of the interleaved $\Delta\Sigma$ DAC increases by $20 \log_{10} P \ dB$ due to the coherent summation of the signal while the noise density increases by $10 \log_{10} P \ dB$. The difference between the increase in signal power and noise density causes an increase in the resolution of the interleaved architecture by $0.5(P - 1)$ bits [27].

### 3.2.2 Inter-DAC Amplitude and Timing Errors

The rejection of unwanted images by the interleaved DSM and DAC is heavily dependent on the accuracy of the channel D/A conversion and output summation. Depicted in Fig. 3.9(d), variations in signal amplitude or timing between the interleaved paths results in non-ideal image suppression. This non-ideality is quantified by the signal-to-image replica ratio (SIRR), expressed in (3.12) as the difference between the signal power and the suppressed image power. Based on the similarity observed in [65] between the interleaved-by-two architecture and an image reject mixer [66], (3.12) takes into account the error in timing, $\delta t$, between the 180° system clocks as well as the amplitude ratio, $\gamma$, between the two DAC outputs (ideally $\gamma = 1$). The values $f_0$ and $f_I$ are once again the signal frequency and the image frequency, respectively.

$$SIRR = 10 \log \left( \frac{\gamma^2 + 1 + 2\gamma \cos(\pi f_0 \delta t)}{\gamma^2 + 1 - 2\gamma \cos(\pi f_I \delta t)} \right) \ [dB]$$  \ (3.12)

In (3.12), it is assumed that the signal and image are frequency adjacent ($f_0 \approx f_I$) and both undergo the same hold-shaping. While this is a good approximation in the HP case since $f_0$ and $f_I$ are within the same passband, signals and images with a significant frequency offset experience different magnitudes of hold shaping. To
account for this, the ratio between the signal and image shaping is calculated in (3.13) and applied as a correction to the SIRR value in (3.14); a DAC ZOH is assumed.

\[ S_{ZOH} = 20 \log \left( \frac{f_l \sin (\pi f_0 / f_S)}{f_0 \sin (\pi f_l / f_S)} \right) = 20 \log \left( \frac{f_l}{f_0} \right) \text{ [dB]} \]  
\hspace{1cm} (3.13)

\[ S_{IRR_{ZOH}} = S_{IRR} + S_{ZOH} \]  
\hspace{1cm} (3.14)

To verify this expression, Fig. 3.11 shows a simulation\(^4\) of the interleaved architecture with amplitude and timing errors injected into one of the interleaved channels. In this simulation, a variety of error values are simulated for both types of systematic mismatch and the resulting outputs are overlayed. As before, these simulations assume second order HP DSMs with 3-bit DACs, the LSB current of each set to 1 mA.

For comparison, the simulated SIRR values from Fig. 3.11 are plotted in Fig. 3.12(a) alongside the values calculated by (3.14). The same is also done for a BP-configured DSM with the results compared in Fig. 3.12(b). In both cases, the simulated values confirm the validity of the analytical model which describes the system performance under systematic amplitude and timing errors. A comparison with the LP-configured DSM is not shown due to the high attenuation of the ZOH shaping for this passband.

\(^4\)Refer to Appendix A
Figure 3.11: Simulation of a second order interleaved HP DSM and DAC (3-bits with 1 mA $I_{LSB}$) showing the SIRR for systematic (a) amplitude and (b) timing errors between the two DACs. The error magnitudes of the overlayed simulations are given alongside each SIRR.
Figure 3.12: A comparison between the calculated and simulated SIRR values for both (a) the HP-configured DSM and (b) the BP-configured DSM. The percent error of the timing mismatch is given relative to $f_S$. 

$f_0 = 0.49 f_S$

$f_0 = 0.75 f_S$
3.2.3 Intra-DAC Errors

In addition to systematic error between channels, the interleaved design can also experience amplitude and timing errors within each DAC. In addition to limiting the SIRR, these non-idealities not only limit SIRR, they also result in non-linear DAC outputs which reduce the DAC spurious-free dynamic range (SFDR), increase inter-modulation tones, and corrupt the ΔΣ passband noise-floor. These effects can also be simulated with the model in Appendix A. Figure 3.13 demonstrates the performance degradation due to mismatch errors of 0.0%, 0.3%, and 3.0% within one of the interleaved DACs. As shown, the DAC images reappear due to imperfect DAC matching while higher harmonics and noise are folded in-band due to the non-linear DAC operation.

![Figure 3.13: Simulated spectrum with varying current-cell mismatch within one DAC.](image)

Figure 3.13: Simulated spectrum with varying current-cell mismatch within one DAC.
3.3 Comparison to Nyquist-Rate Architecture

A precise comparison between the proposed \(\Delta\Sigma\) DAC and a Nyquist-rate mixing DACs is critical to the understanding of the trade-offs between the two architectures. Due to the oversampling and noise shaping of the \(\Delta\Sigma\) DAC, a comparison of the architectures across a large bandwidth is difficult. Instead, the metrics of in-band SNR and noise power-density are utilized for their relevance to RF signal synthesis.

3.3.1 In-band SNR and ENOB

For a Nyquist-rate DAC (Fig. 3.14(a)), an increase in the number of bits, \(N\), reduces the total quantization noise power throughout the Nyquist band. Additionally, oversampling can be applied by increasing the sampling frequency, \(f_{SN}\), above the Nyquist rate to further improve the SNR by spreading the quantization noise outside of the desired signal band. The in-band SNR of a Nyquist-rate DAC is given by [67]

\[
SNR_N = 6.02N + 1.76 + 10\log_{10}[OSR_N] \text{ [dB] (3.15)}
\]

where

\[
OSR_N = \frac{f_{SN}}{2f_B} \quad (3.16)
\]

In contrast to Nyquist-rate baseband or mixing DACs, the in-band SNR of the proposed \(\Delta\Sigma\) DAC (Fig. 3.14(b)) is determined by several factors: \(M, N, OSR_{\Delta\Sigma}\), and \(\alpha\). Assuming a dual Nyquist zone output and an \(OSR_{\Delta\Sigma} \gg 1\), the shaped in-band noise power of an ideal interleaved-by-two LP or HP DSM and DAC is expressed in [59]

\[
q_{rms, LP}^2 = q_{rms, HP}^2 = \frac{\pi^{2\alpha}\varepsilon_{rms}^2}{2^{2\alpha - 1} (2\alpha + 1) OSR_{\Delta\Sigma}^{2\alpha + 1}} \quad (3.17)
\]

55
Increasing OSR
Increasing LPF
Increasing Sample-Rate
Increasing Resolution
Shaped Noise
DAC
LO
LPF
SN
(a)
DAC Output
0
f_{SN1}/2
f_{SN2}/2
f_{SN1} < f_{SN2}
(b)
DAC Output
0
f_{S1}/2
f_{S2}/2
f_{S1} < f_{S2}

Figure 3.14: Analog synthesis and up-conversion of a signal bandwidth $f_B$ showing the effects of increased resolution and sample-rate for (a) a baseband Nyquist-rate DAC and (b) the proposed $\Delta\Sigma$ RF-DAC.

where

$$OSR_{\Delta\Sigma} = \frac{f_{S_{\Delta\Sigma}}}{2f_B}$$  \hspace{1cm} (3.18)

For the BP-configured DSM, the shaped in-band noise power is given in (3.19) [68].

$$q_{rms,BP}^2 = 2^{2\alpha+1} \frac{\pi^{2\alpha} \varepsilon_{rms}^2}{(2\alpha + 1) \cdot OSR_{\Delta\Sigma}^{2\alpha+1}}$$ \hspace{1cm} (3.19)

Note that the passband for a BP DSM is located in the middle of the Nyquist zone such that multi-Nyquist operation does not impact the in-band noise power. The quantization noise at the input of the $\Delta\Sigma$ DAC due to finite $M$ is added to the shaped quantization noise to yield a total in-band noise power of

$$Q_{rms}^2 = q_{rms}^2 + \frac{\Delta_N^2}{6(2^{M-N})^2} \cdot \frac{1}{OSR_{\Delta\Sigma}}$$ \hspace{1cm} (3.20)

The signal power of the interleaved-by-two $\Delta\Sigma$ DAC is given in (3.21) and is derived from the maximum DSM input range $x_{MAX}$, previously defined in (3.8), and the
quantization bin width $\Delta_N$.

$$X_{\text{rms}}^2 = \frac{\Delta_N^2 \cdot x_{\text{MAX}}^2}{2} \quad (3.21)$$

Combining (3.21) and (3.20) yields the in-band SNR of the $\Delta\Sigma$ DAC as expressed in (3.22).

$$SNR_{\Delta\Sigma} = 10 \log_{10} \left[ \frac{X_{\text{rms}}^2}{Q_{\text{rms}}^2} \right] \quad \text{[dB]} \quad (3.22)$$

For a synthesized bandwidth, $f_B$, a Nyquist-rate mixing DAC and the proposed $\Delta\Sigma$ DAC can be compared by relating both to the equivalent number of bits (ENOB) in a critically-sampled converter, as given in (3.23) [67].

$$ENOB = \frac{SNR - 6.02}{1.76} \quad \text{[bits]} \quad (3.23)$$

A Nyquist-rate mixing DAC is theoretically interchangeable with a $\Delta\Sigma$ DAC over a given bandwidth if both DACs exhibit a similar ENOB. Combining (3.15) and (3.22) into (3.23), the resulting ENOB is shown in Fig. 3.15 for both a Nyquist-rate and an HP-configured $\Delta\Sigma$ DAC with varying values of $N$ and $\alpha$ across $OSR^6$. As shown, Nyquist-rate mixing DACs benefit more from increasing $N_N$ than from increasing $OSR_N$. As a result of the difficulty in achieving sufficient linearity with high $f_{SN}$ and large $N_N$, such systems have historically been limited to a few GHz, effectively limiting $OSR_N$ [11, 12, 24]. In contrast, $\Delta\Sigma$ DACs can achieve a similar ENOB as Nyquist-rate DACs with fewer bits at the output ($N \ll N_N$), facilitating the use of higher sampling rates and thus larger OSRs [20, 22, 26, 30]. To illustrate the effect of this difference in OSR, sampling rates of $f_{S_{\Delta\Sigma}} = 4f_{SN}$ and $f_{S_{\Delta\Sigma}} = 8f_{SN}$ are used in Fig. 3.15(d) to compare the ENOB of a $\Delta\Sigma$ DAC to that of a Nyquist-rate DAC ($OSR_N = OSR_{\Delta\Sigma}/4$ and $OSR_N = OSR_{\Delta\Sigma}/8$). A $\Delta\Sigma$ DAC with $\alpha = 2$ 

\footnote{To prevent overflow, $N \geq \alpha$.}

\footnote{The $Q_{\text{rms}}^2$ approximation assumes $OSR_{\Delta\Sigma} \gg 1$.}
Figure 3.15: Theoretical performance of (a) the oversampled Nyquist-rate DAC with varying number of bits $N_N$, (b) a multi-Nyquist LP or HP $\Delta\Sigma$ DAC with $\alpha = 2$, (c) $\alpha = 3$, and (d) comparison of the $\Delta\Sigma$ DAC and a 14-bit Nyquist-rate DAC with $OSR_N = OSR_{\Delta\Sigma}/4$ and $OSR_N = OSR_{\Delta\Sigma}/8$.

can theoretically exceed the performance of a 14-bit Nyquist-rate DAC given that $OSR_N = OSR_{\Delta\Sigma}/4 > 40$ or $OSR_N = OSR_{\Delta\Sigma}/8 > 30$. For $\alpha = 3$, an $OSR_{\Delta\Sigma} > 10$ is sufficient to exceed the Nyquist-rate DAC performance for $OSR_{\Delta\Sigma}/4 > OSR_N$.

From this analysis, it is evident that the ENOB (i.e. the noise floor performance) of the $\Delta\Sigma$ DAC can exceed that of a Nyquist rate design within a given RF passband. While this analysis does not take into account jitter or non-linearity when estimating
ENOB, it can be assumed that a lower resolution design \( N < N_N \) will be smaller and experience less non-idealites.

### 3.3.2 Noise Floor

In addition to the in-band ENOB, a comparison between the in-band noise power-density of the \( \Delta \Sigma \) DAC and that of the Nyquist-rate mixer DAC can be made. Equation (3.24) gives the noise PSD of the proposed architecture at any frequency \( f \).

In this equation, both the noise shaped by the DSM and the unshaped input noise are considered. The expression \( |1 - G(f)| \) represents the magnitude of the reconfigureable noise transfer functions where \( G(f) \) is one of the filter functions given previously in (3.5)-(3.7) and where \( z^{-1} = e^{-j2\pi f/f_{\Delta \Sigma}} \). Substituting (3.21) and (3.24) into (3.25) gives the noise power-density in dBc/Hz.

\[
NF(f) = |1 - G(f)|^2 \left( \frac{2\varepsilon^2_{\text{rms}}}{f_{\Delta \Sigma}} \right) + \frac{\Delta^2_N}{6(2^{M-N})^2} \left( \frac{1}{f_{\Delta \Sigma}} \right) \quad (3.24)
\]

\[
NF_{dBc}(f) = 10 \log_{10} \left( \frac{NF(f)}{X^2_{\text{rms}}} \right) \quad \text{[dBc/Hz]} \quad (3.25)
\]

Fig. 3.16 plots (3.25) for both HP and BP-configured DSMs, comparing their in-band noise performance to an ideally up-converted 14-bit Nyquist-rate DAC. For the \( \Delta \Sigma \) DAC, variations in \( \alpha \) are plotted with \( f_{\Delta \Sigma} = 2 \) GHz and \( N = 3 \) bits, while \( f_{\Delta \Sigma} = 4f_N \) and \( f_{\Delta \Sigma} = 8f_N \) are plotted for the Nyquist design. Due to the noise shaping and the typically higher oversampling of the DSM, the proposed architecture achieves a much lower in-band noise PSD than its Nyquist-rate counterparts, meeting ultra-low in-band noise specifications at the expense of out-of-band noise filtering.
Figure 3.16: The noise PSD of the proposed ΔΣ DAC for (a) a HP-configured DSM and (b) a BP-configured DSM in the 2nd Nyquist zone (f_{\Delta\Sigma} = 2 GHz and N = 3). The noise PSD of an ideally up-converted 14-bit Nyquist DAC is also plotted for comparison (4f_{SN} = f_{\Delta\Sigma} and 8f_{SN} = f_{\Delta\Sigma}).

Another important performance metric of the proposed architecture is the average in-band PSD of the DAC output, found in (3.26) by taking the mean noise PSD of the output over bandwidth f_B.

\[
NF_{Avg,\text{dBc}} = 10 \log_{10} \left( \frac{Q_{rms}^2}{f_B \cdot X_{rms}^2} \right) \quad \text{[dBc/Hz]} \quad (3.26)
\]

This average power density mirrors the trends in Fig. 3.15 and is plotted for varying \(\alpha\) and OSR in Fig. 3.17(a). The bandwidth is set to 50 MHz to give absolute noise-floor values. The performance of a 14-bit Nyquist-rate converter is plotted alongside that of the proposed architecture, again showing the benefit of the ΔΣ DAC for increasing OSR and \(\alpha\). Additionally, from this plot we can determine the OSR_{\Delta\Sigma} which gives an equivalent average noise-density to that of a Nyquist-rate converter. These results are summarized in Fig. 3.17(b) which shows the oversampling ratio.
Figure 3.17: (a) The average in-band PSD of the proposed ∆Σ DAC compared with that of a 14-bit Nyquist-rate design for $f_B = 50$ MHz. In the plot, $OSR_N = OSR_{\Delta \Sigma}/4$ and $OSR_N = OSR_{\Delta \Sigma}/8$. (b) The $OSR_{\Delta \Sigma}$ needed to achieve equivalent in-band noise performance to that of an $N_N$ resolution Nyquist-rate converter.

required to achieve the equivalent noise floor performance of an $N_N$-bit Nyquist-rate DAC.

3.3.3 ∆Σ Output Efficiency

In addition to the ENOB and noise floor of the proposed ∆Σ DAC, the efficiency of the architecture can also be compared to that of a Nyquist-rate design. To do this, the DAC is assumed to consist of several Class A current cells of various weights, arrayed to synthesize waveforms from an incoming bit stream. Current sources in each cell set the current swing of each cell with the output signal realized across a pair of load resistors. As the current of each cell is increased (and consequently the output power) the output impedance of each current source is reduced, limiting the overall DAC linearity. Cascoding is often used to raise current source output
impedance at the expense of increased voltage overhead. In order to analyze the impact of this overhead on efficiency, the Class A current-steering DAC model in Fig. 3.18 is adopted. Two resistors \( R_L \) represent the DAC output load and ideal current sources model the transistor network, modulating the output current, \( I_Q \). The voltage \( V_Q \) is the potential across the transistor network with \( V_{QMIN} \) being the minimum overhead required for linear operation. Equation (3.27) relates the Nyquist DAC drain efficiency to the DAC LSB current \( I_{LSB} \), DAC resolution \( N_N \) bits), and \( V_{QMIN} \).

\[
\eta_{\text{ClassA}} = \frac{I_{LSB}R_L(2^N - 1)}{4(I_{LSB}R_L(2^N - 1) - V_{QMIN})} 
\]  

(3.27)

The relationship shown in (3.27) demonstrates that the ideal efficiency of a current steering DAC is 25% with a resistive load (50% with an inductive load) assuming no
Figure 3.19: Comparison of (a) Class A DAC and (b) ∆Σ DAC efficiency across $V_{QMIN}$.

voltage overhead is required ($V_{QMIN} = 0$). Note that the sampling and frequency dependent effects of ZOH spectrum shaping are not included in this analysis.

As previously discussed, applying ∆Σ modulation to the DAC input reduces the number of DAC current cells while enabling low in-band noise. In this scheme, however, the time-domain results of noise shaping must consume a portion of the output range, limiting the effective signal amplitude. The voltage overhead required for quantization noise shaping is directly related to the amplitude backoff value given in (3.8). Applying (3.8) to (3.27) yields the drain efficiency of the ∆Σ DAC in (3.28).

$$\eta_{\Delta \Sigma} = \frac{I_{LSB}R_L (2^N - 1) x_{MAX}^2}{4 (I_{LSB}R_L (2^N - 1) - V_{QMIN})}$$  \hspace{1cm} (3.28)

Figure 3.19 illustrates the efficiency of both the Class A (a) Nyquist and (b) ∆Σ DACs. As expected, the efficiency of the Nyquist rate design decreases with increasing $V_{QMIN}$. In the ∆Σ design, efficiency approaches that of a Nyquist DAC as quantization bits ($N$) increase (due to reduced overhead required for noise shaping) or the modulator order decreases (reducing the gain of the noise shaping).
While Class A current-steering architectures remain the preferred choice for high linearity DACs and modulators, both Class A designs perform poorly when efficiency is considered (< 25%). The impact of \( V_{QMIN} \) on efficiency favors the reduced overhead of switched mode designs over saturated transistor architectures. This can be accomplished when the triode conductance of the transistor is utilized to create the cell current instead of the device transconductance. However, as explain in Chapter 5.1.3, this choice would limit the output impedance of the overall design, leading to a trade-off between efficiency and linearity similar to that seen in PA design. Given the stringent IM3 and ACLR requirements of modern communication systems, the high-linearity Class A topology is favored over that of a more efficient design.
Chapter 4: $\Delta \Sigma$ DAC - First Revision

The goal of the initial RF-DAC implementation is singular: the verification of the $\Delta \Sigma$ interleaving proposed in Chapter 3. To this end, the first-pass design is kept simple, relying on conventional circuit building-blocks to prove the concept. With the design experience and testing results of this revision, subsequent versions of the proposed RF-DAC can be further optimized for RF performance. This section outlines this design and its testing, concluding with a discussion of the measurement results.

4.1 Circuit Design

While the focus of this design is the validation of the concepts in Chapter 3, the infrastructure needed in support of this goal is also critical to the success of this endeavor. For this reason, both the DAC design and other supporting structures are discussed herein.

4.1.1 DAC and Data Retiming

Any implementation of the proposed RF-DAC must have as its first objective the elimination of both inter-DAC timing ($\delta t$) and amplitude ($\gamma$) variations so as to maximize the DAC SIRR. To achieve this, two identical DACs with matched clock
distribution and data retiming comprise the core of the design, as shown in Fig. 4.1(a). Additionally, the identical (unary weighted) DAC current cells in Fig. 4.1(b) are implemented to further reduce susceptibility to process mismatch [69]. For Nyquist DACs, this design choice typically leads to a large layout area, increasing timing errors. The proposed architecture avoids this by limiting the ΔΣ RF-DAC to $N = 3$ bits.

In addition to these design choices, the current cells in each DAC utilize the available BiCMOS process to improve high-speed performance. The CMOS transistor $M_1$ is used as a current source for its low voltage overhead, minimizing power dissipation. Cascoding with $M_2$ improves the linearity of the DAC by increasing the output resistance of the current source, reducing drain-induced current errors and data-dependent glitches. The data-switching pair, $Q_1$ and $Q_2$, are implemented with SiGe HBTs. Their high $f_T$ ensures the cell current is fully commutated within one clock period. An additional cascode pair, $Q_3$ and $Q_4$, increases the output impedance of the current cell, reducing DAC output impedance variations. The SiGe HBTs exhibit superior output impedance and switching speed compared to their 130 nm CMOS counterparts, making them the optimum choice for the DAC data path.
Figure 4.1: (a) The block diagram of the interleaved RF-DAC. The (b) schematic of DAC0 and (c) the diagram of one LVDS channel.
4.1.2 Data I/O and Loopback Testing

The previously designed LVDS receiver, pictured in Fig. 4.1(c), enables the high-speed input of BP and HP ΔΣ modulated data into the DAC. Given the 3-bit resolution of each interleaved DAC, only six LVDS receivers are needed to feed the design. With its small on-chip capacitor, the design replaces several bulky, on-board capacitors with fully integrated ones consuming less than 0.06 mm². As detailed in Chapter 2, the AC coupled RZ input of the pulse receiver minimizes baseline wander and requires no data encoding. This simplifies the DAC test-setup requirements, enabling a singular focus on the designs interleaved performance. In addition to the pulse receivers, the transmitter used for LVDS loop-back testing is also included alongside the DAC. In this way, the on-chip data latching of the design can be tested separately from the DAC, allowing the design to be validated in stages and aiding any debugging procedures.

4.2 Measurement

As with the LVDS receiver, the first revision of the proposed ΔΣ RF-DAC is fabricated in a 0.13 μm SiGe BiCMOS technology. Interleaved clock generation, clock routing, and both 3-bit DACs are implemented on-chip while the ΔΣ modulation is located off-chip, accomplished via pre-compiled ΔΣ data created within Matlab. For testing, this data is loaded into an FPGA and fed into the design through the previously-verified LVDS receivers. From this architecture, the intrinsic amplitude and timing accuracy of the design can be tested without the added overhead of an on-chip DSM. Additionally in this revision, the inclusion of calibration circuitry is foregone in favor of a simple first proof-of-concept. This test-setup is depicted in
Figure 4.2: Test setup of the first $\Delta\Sigma$ RF-DAC revision.

Fig. 4.2, showing the data generation, FPGA input, and DAC functionality of the design.

### 4.2.1 Test Infrastructure

The flip-chip die (Fig. 4.3) is mounted onto a custom 4-layer PCB (Fig. 4.4). This board mediates all chip connections; the LVDS data is brought in from the east and west, clock input from the south, and the DAC output is routed north. Due to limited board space, the loopback clock and data are routed vertically off the board. Chip voltages are connected through on-board header pins and careful attention is paid to decoupling. Large ($\mu F$) decoupling capacitances are used at the header inputs and smaller capacitances ($nF$ and $pF$) are located next to the chip for close-in voltage conditioning. For signal integrity, all high-speed lines are routed on the top layer with 50 $\Omega$ lines. Internal layers are used for high capacitance power routing and the bottom layer carries low frequency signals and biases. Input current biases are created from
Figure 4.3: Photograph of the first $\Delta \Sigma$ DAC prototype showing LVDS input, clock generation, and the $0.18 \text{ mm}^2$ DAC core.

High-precision potentiometers (labeled) and switched in via SPDT DIP switches. Off board, the LVDS lines are connected to a VC707 Virtex 7 evaluation board on which second-order $\Delta \Sigma$ modulated data is loaded and fed into the design.

As shown in Fig. 4.2, the $180^\circ$ clocks necessary for interleaving are generated by dividing the off-chip reference. Coarse timing adjustment is available on-chip to ensure accurate data/clock alignment. However, phase accuracy between the two clocks ($\delta t$ in (3.12)) is determined by layout symmetry. One available adjustment is the LSB current of each DAC, which can be set independently, allowing for manual tuning of $\gamma$. At the DAC output, the interleaved design is connected to an on-board bias-T to separate the input DC bias from the output RF which is measured with an Agilent PXA spectrum analyzer.
4.2.2 Results

Figure 4.5(a-b) demonstrates all three configurations of the proposed RF-DAC: (a) BP outputs at either $f_s/4$ or $3f_s/4$ and (b) a HP output at $f_s/2$. Evident in the plots is the distinctive noise shaping of the DSMs and the ZOH sinc response, attenuating the $f_s/2$ output by $\approx 3\, \text{dB}$ and the $3f_s/4$ output by $\approx 10\, \text{dB}$. At $f_s = 2\, \text{GHz}$, the integrated DAC core consumes a total of 55 mW from a 3.3 V source.

In Fig. 4.5(c), the HP output spectrum around $f_s/2$ is given for both a single and interleaved DAC. Since the signal lies at the Nyquist boundary in this $\Delta\Sigma$ configuration, a single DAC produces an in-band image (red). For the interleaved case (black),
an SIRR of 65 dB is measured without calibration, validating the intrinsic matching
between the two DACs and within the clock routing. Independently adjusting $I_{\text{LSB0}}$
improves gain matching, bringing the SIRR up to 72 dB. With $\gamma$ now assumed ideal,
the SIRR measurement corresponds to an inter-DAC timing error of $\delta t < 80 \, f s$.

While the inter-DAC gain and phase error determine the SIRR, intra-DAC mis-
matches set the in-band SFDR and noise performance (previously shown in Fig. 3.13).
In the implementation of the proposed $\Delta \Sigma$ RF-DAC, a linearity of 99% is obtained
(Fig. 4.6) while an SFDR of 58.5 dB and an SNR of 54.5 dB are achieved over a band-
width of 50 MHz. The SNR performance, as well as the output frequency, output
power, and DSM type are compared to the second prototype and to other published
designed in Table 5.2.
Figure 4.5: Measured spectrums at $f_s = 2$ GHz. (a) BP and (b) HP outputs from 0-2 GHz. (c) Close-in HP spectrum showing both single DAC and interleaved outputs. (Not de-embedded)
4.2.3 Take Aways

The test results of the first prototype verify both the theory behind the interleaved $\Delta\Sigma$ architecture as well as its practical implementation. Without the aid of on-chip calibration, the system SIRR is well in excess of $60\,dB$, pushed above $70\,dB$ with systematic DAC amplitude adjustment. However, the IM3 and SFDR of the design are both below $60\,dB$, requiring improvement in the second revision of the DAC. For the low SFDR depicted in Fig. 4.5, the degradation is a result of a third harmonic tone folding back in-band. This tone is traced back to deterministic jitter in the retiming FF, alleviated in the subsequent revision by the CML architecture discussed in Chapter 5.1.3. The poor IM3 of this design arises from static current-cell nonlinearities, removed by amplitude calibration in the second iteration. Additionally, the marginal SNR performance arises from stochastic jitter in the full-scale (inverter)
clock drivers, remedied with high-power, low-jitter CML clock drivers. Also of note in the first DAC revision is the low (≈ 80 fs) timing error attainable through careful on-chip layout. This error value is later used in the design of the second revision timing calibration (Chapter 5.1.3), ensuring a minimum SIRR performance equivalent to that of the first revision.
Chapter 5: $\Delta \Sigma$ DAC - Second Revision

Given both the success and the shortcomings of the first revision, the second DAC prototype adds additional capability to the design, such as an on-chip DSM and per-cell DAC calibration, as well as remedy several issues which effected the SFDR, IM3, and SNDR performance. This revision (Fig. 5.1) is comprised of two parallel data paths, each aligned to separate 180° clocks derived from an off-chip 2x source. Interleaved samples are received from an FPGA through seven LVDS channels parallelized into two 14-bit buses to accommodate the data rate required by the now on-chip DSM. Incoming data is offset off-chip to account for the pipelining of the DSM computation. A data path bypassing the DSM is also implemented to enable stand-alone DAC testing (as in the first revision) and calibration. After $\Delta \Sigma$ modulation, the 3-bit DSM outputs are thermometer encoded, re-timed by clean clocks, and driven into the DAC current cells. To ensure accurate interleaving and linear DAC performance, fine amplitude and timing adjustments are also implemented via SPI register control.
5.1 Circuit Design

With the increased size and complexity of the second DAC, several new circuits are included on-chip. These are highlighted in the following section with special attention payed to the redesigned data retiming and DAC core.

5.1.1 Serial Data Input

Since the on-chip ΔΣ function operates at the DAC GS/s rate, the architecture requires significant I/O bandwidth for each interleaved 14-bit data stream. To meet this requirement, the low-power LVDS receivers discussed in Chapter 2 are implemented on-chip. Despite the high-speed capability of the asynchronous LVDS receivers (10 Gb/s), the speed through each channel is limited due to the need for
alignment and sampling of the incoming data. Because of this, the data I/O is designed to receive at least a 4 Gb/s double-data rate input stream, enabling fourteen 2 Gb/s data input channels with 7 receivers per DAC. In addition, three LVDS channels can be configured to bypass the DSM for standalone DAC testing. As before, the receiver is AC coupled on-chip with 1.4 pF capacitors, minimizing layout area. Following the receiver, incoming data is passed through a 16-tap adjustable delay line to compensate for off-chip clock-to-data skews of up to 1.2 ns, as shown in Fig. 5.2. Once data is correctly aligned to the on-chip clock, the 4 Gb/s data from each channel is latched and separated into two 2 Gb/s streams via two flip-flops (FF) operating on opposite clock edges.

5.1.2 ΣΔ Modulator

Given the fundamental link between sample rate and bandwidth in ΣΔ modulation, a significant focus of recent DSM work has been the high-speed (GS/s) operation of DSMs, achieved through a combination of novel digital architectures [18, 20], phase
unrolling [70], pipelining [71], and time interleaving [29, 30]. However, existing architectures target baseband synthesis, utilizing LP DSMs to generate near-DC signals. In contrast, this chip implements a time-interleaved design which includes both the DSM and DAC, as discussed in Chapter 3. This architecture increases the effective sample rate of the system without requiring a high-speed mux and a subsequent higher-frequency clock domain to reconstruct the modulated signal. The proposed DSM also leverages its all-digital architecture to switch between multiple filter functions and shaping orders.

**Architecture**

For the proposed DSM to switch between multiple filter functions while achieving RF sample rates, a MASH-11 architecture, depicted in Fig. 5.3, is used to pipeline the order of the design. Two first order stages, consisting of single tap filters (5.4(a)), reduce feedback latency compared with a higher-order, single stage DSM. As in 3.3, the first MASH stage shapes the quantization error $\varepsilon_1[n]$. This error is also passed to the second stage whose output is given by

$$y_2[n] = \varepsilon_1[n] + \varepsilon_2[n] (1 - G(z))$$  \hspace{1cm} (5.1)

Multiplying this output by $H(z)$ allows $y_2[n]$ to be scaled such that $\varepsilon_1[n]$ is completely canceled in the DSM output summation and the remaining error $\varepsilon_2[n]$ is shaped with a higher order NTF [59]. By setting $H(z) = G(z) - 1$, the MASH output becomes

$$y[n] = x[n] - \varepsilon_2[n] (1 - G(z))^2$$  \hspace{1cm} (5.2)
Switching between the second stage output and zero, as depicted by the output combining in Fig. 5.3, allows the design to be configured for first or second order shaping. To further maximize the speed of the modulator, each MASH stage is itself pipelined using 1-bit unit cells. In this way, the logic operation performed during a single clock period is reduced to that of a 1-bit subtractor with the borrow propagated via a ripple architecture. A total of eleven of these cells [0:10] are used to shape the LSB quantization error in each MASH stage while MSB cells account for any borrow signal emerging from the noise shaping.

Within the LSB cells and the output combining, reconfigurable feedback filters are used to realize both BP and HP shaping. Since the desired filter functions differ by a single delay, as highlighted in Chapter 3.1.2, implementing both simply requires alternating between two delay values (Fig. 5.3(b)). Altogether, the pipelined structure allows the proposed DSM to synthesize the outputs depicted to Fig. 5.3(c): first and second order shaping of both BP and HP passbands. While this architecture requires staggered data input and several alignment stages, increasing the power, area, and latency through the design, it also creates a robust, highly modular architecture based on the design of a single bit cell.
Figure 5.3: MASH-11 modulator pipelined with 1-bit unit cells. The bit cells and MASH stages can be configured for BP or HP outputs and for first and second order shaping.

Figure 5.4: (a) Two reconfigurable bit cells, and (b) multimode ΔΣ shaping with both the signal and first image shown.
DSM Bit Cell

As previously mentioned, the 14-bit on-chip DSM is pipelined using 1-bit unit cells. This approach allows an increase of the DSM speed up to the latency of the feedback (FB) path. Shown in Fig. 5.5, this critical path consists of only the subtract logic and the HP/BP switches. For low latency, the subtraction is computed with an optimized 1-bit ripple carry add [72] with the borrow logic implemented in parallel. In the subtraction, the final inversion of $\bar{S}$ to $S$ is relocated from directly after the function (grey) to after the delay FFs, allowing the BP and HP switches to be implemented with tri-state inverters (black) instead of slower, un-driven transmission gates (grey). Minimum area and power true single phase clock (TSPC) FFs [73] are driven by the tri-state inverters, reducing the FB latency by 40 ps in simulation, increasing the timing margin by 8% of the clock period at 2 GS/s.

Figure 5.5: The single bit feedback cell used to pipeline the proposed reconfigurable DSM and showing the relocation of the $\bar{S}$ inversion.
5.1.3 Data Path and DAC

Due to the digital nature of the DSM, the noise shaping and linearity of the modulator output are ideal. However, the subsequent DAC core limits the fidelity of the RF output, directly determining the SFDR and SNDR performance of the RF signal. This requires a DAC core design which achieves much higher linearity than a standard 3-bit Nyquist design. Shown in Fig. 5.6, the core of each interleaved DAC is comprised of seven unary-weighted current switching cells. The unary architecture exhibits low data-dependent glitches at the DAC output and relaxes the constraints on current matching compared to the conventional binary and segmented schemes [69, 74].

DAC Current Cells

The proposed design uses SiGe HBTs, mitigating nonlinearities caused by data-dependent impedance modulation [75]. Though SiGe HBTs are large, the DAC core comprises only seven unary cells allowing for a compact layout with reduced routing parasitics [76] and timing skew [77].

The current source in the DAC cell is implemented via CMOS transistors and placed outside the DAC cell array for compact interdigitation and optimized matching as shown in Fig. 5.6. The output of each current source is routed to an individual cell, where it is cascoded with an HBT just before the data switches, providing high output impedance for the switched pair by shielding them from the interconnect capacitance. The data switches are implemented using HBTs to provide superior switching speed and output impedance as compared to their CMOS counterparts. Additionally, cascoded HBTs shield the data switches from the DAC output, further
increasing the cell output impedance. A CMOS bleed current (35 $\mu$A) is added to each cascode to improve SFDR by keeping the cascode transistors “on”, thereby reducing the data-dependency of the cell output impedance and reducing the settling time when the corresponding data switch is turned off [17].

Figure 5.6: Schematic and layout of the data path and DAC current cells highlighting the minimized deterministic jitter.
Data Path

To maximize the linearity and noise performance of the DAC current cell, any data dependent timing effects on the data inputs to the cell must be minimized. While the impact of deterministic jitter has been analyzed in Nyquist DACs [78], to the author’s knowledge, no such analysis has been published for ∆Σ DACs. Figure 5.7(a) shows behavioral simulations of the DAC in the presence of deterministic jitter at the input to each current cell. A peak-to-peak deterministic jitter of just 1% of the clock period (3.3 ps at 3 GS/s) results in degradation in SFDR and SNR of nearly 20 dB and 6 dB, respectively. To minimize this effect, the data path, shown in Fig. 5.6, utilizes CML structures to re-time the data and drive appropriate levels to the data switches in the current cell. Full-scale CMOS data from the thermometer encoder drives the first CML latch. While this stage removes much of the data dependent jitter induced by the encoder and preceding circuitry, some remains (≈ 130 fs of p-to-p jitter) due to the unbalanced differential input created by the inverter delay and due to the CMOS inputs overdriving the CML latch, pushing it out of saturation. The remaining jitter is nearly eliminated by a second CML latch that is driven with all CML signals. The re-timing latches are followed by two stages of CML buffers. These buffers slightly increase the deterministic jitter because they are not clocked, however, they reduce the clock feedthrough from the latches and provide a voltage swing that is designed to minimize data-dependent jitter in the current cell switches [79]. Supply voltage induced deterministic jitter is mitigated by separating the power supply for the re-timing data path and clock drivers from the rest of the digital circuitry on the chip. Figure 5.6 shows the simulated eye diagram at the outputs of the second data driver.
Amplitude and Timing Calibration

Even with the size and matching benefits of a low-bit ΔΣ DAC, the proposed multi-bit design ($N = 3$) still faces stringent requirements on linearity and timing to ensure that the rejected image and other spurious tones do not degrade the SNDR of the system. Static current mismatch within one DAC can manifest as spurious tones in the output while current mismatch between the interleaved DACs decreases the effectiveness of image cancellation, as described in (3.12). A Monte Carlo (MC) simulation performed on the interleaved ΔΣ DACs identifies the impact of static current mismatches, yielding the SFDR (including the replica image spur) in Fig. 5.7(b). Each MC run includes 1500 iterations, and the 97.7% yield line is plotted using a 95% confidence level. To ensure an SFDR above 70 dB, a mismatch tolerance of $\sigma_I/I_{\text{LSB}} = 0.03\%$ is required. To achieve this in a reasonable area, the CMOS current
source is designed for a 0.3% mismatch, and a 9-bit calibration DAC is used to fine tune the current of each cell. Similarly, static timing mismatches in the DACs can also degrade the SFDR and SIRR. Fig. 5.7(c) gives the MC results of DAC SFDR for various static timing mismatches between cells. As shown, a timing mismatch of 0.013% is required to achieve 70 dB of SFDR. To account for this possible mismatch, a 5-bit timing adjustment circuit is implemented via the interpolative clock buffer shown in Fig. 5.8. In this design, $CLK_{IN}$ and $\overline{CLK}_{IN}$ are separately weighted by the current adjustment and then summed at the output node, allowing the design to shift the clock in $\approx 90pS$ steps across a range of $< 2.8ps$.

**Clock Jitter**

While mismatches dramatically affect the DAC SFDR, they have less impact on in-band noise performance. This is highlighted in Fig. 5.7(b-c) where the $\Delta \Sigma$ DAC SNR is plotted with respect to MC mismatch for a 5% bandwidth $(0.05 \cdot f_s)$. However, dynamic effects, such as clock jitter, significantly affect SNR performance. To demonstrate this, Fig. 5.8 plots the SNR for two cases: that of global clock jitter (all DAC cells experiencing the same jitter) and local clock jitter (each DAC cell has statistically independent jitter). For low RMS jitter, the SNR of both global and individual clock jitter approach an ideal value set by the noise shaping of the DSM. As jitter increases, the effects of global jitter add coherently and degrade the SNR more severely than that of individual cell jitter, which adds incoherently. This difference is leveraged in the design of the CML clock buffer tree by locating all timing adjustment within each DAC cell, as shown in Fig. 5.8, minimizing its effect on global clock jitter. Additional improvements in SNR can be obtained by allocating more of the clock distribution to local clock buffers, however, this would significantly
Figure 5.8: The DAC clock distribution scheme and simulation of SNR for both local and global drivers.

...increase the power consumption of the chip and is thus avoided in this design. For the implemented 9-stage buffering, extracted simulation estimates a total RMS jitter of 0.63 ps, plotted in Fig. 5.8 for 1 GHz, 2 GHz, and 3 GHz operation. The model used for this simulation is given in Appendix B.

As shown by the preceding analysis, the capability added to the second DAC revision aims to significantly improve the DACs RF performance, leveraging the learning of the first revision to produce a high quality direct digital-to-RF system.
5.2 Measurement

The second revision of the proposed ΔΣ DAC is again fabricated in a C4-bumped 0.13 µm BiCMOS process and packaged flip-chip-on-board for testing. The 3 mm x 5 mm chip, pictured in Fig. 5.9, integrates fourteen LVDS receivers, two reconfigurable DSMs, the CML data path, and the interleaved 3-bit DAC core along with divide-by-two clock generation and digital SPI control, depicted in the functional diagram in Fig. 5.10.

5.2.1 Test Infrastructure

A second board revision builds on the first design, mirroring the on-chip additions of the second chip. Pictured in Fig. 5.11, the board adds additional LVDS inputs, SPI control pins, divided clock output, and more accurate current biasing. The test
setup for this design, shown in Fig. 5.12, utilizes a VC7215 Virtex 7 evaluation board to feed interleaved data samples into the DAC at double the on-chip data rate. DAC waveform data is created from custom Matlab scripts and preloaded into the FPGA. After the D/A operation, the differential DAC output is taken from board mounted SMAs, converted to a single-ended output with a Picosecond Pulse Labs 5310A balun, and measured using a spectrum analyzer. A more detailed overview of the testing procedure is given in Appendix C with the SPI register functions and controls given in Appendix D.
Figure 5.11: Revision 2 test board on which the interleaved DAC is mounted.

Figure 5.12: Test setup of the interleaved ∆Σ DAC.
Figure 5.13: Measured 100 MHz HP passband at 1.0 GHz showing (a) single and (b) two-tone tests as well as the 50 MHz passband at 1.5 GHz with (c) single and (d) two tone tests. The results include a measurement loss of $\approx 7\, \text{dB}$

5.2.2 Results

In Fig. 5.13, the variable shaping of the on-chip DSM is demonstrated at a sample rate of $2\, \text{GS/s}$, showing first and second order shaping for the HP and both BP passbands. These outputs are attenuated by the sinc shaping of the DAC zero-order hold, resulting in a $\approx 10\, \text{dB}$ attenuation of the second BP passband. Ideal second order shaping highlights the in-band noise floor of the design which is limited by the jitter and non-linearity of the DAC, folding noise back in-band. Figure 5.14(a) shows a single tone output spectrum of the DAC at a sample rate of $2\, \text{GS/s}$ and with second order HP shaping. Since the design is implemented without reconstruction filtering, measurements are taken within the $\Delta\Sigma$ passband, a 100 MHz bandwidth (5% of $f_s$). The uncalibrated DAC exhibits an SIRR of 45.9 dB, the image dominating the spurious free dynamic range. To calibrate the DAC amplitude, the output power of
Figure 5.14: Measured 100 MHz HP passband at 1.0 GHz showing (a) single and (b) two-tone tests as well as the 50 MHz passband at 1.5 GHz with (c) single and (d) two tone tests. The results include a measurement loss of $\approx 7 \, dB$

each cell is measured off-chip and set to a common power reference while the timing of each cell is adjusted to minimize the system SIRR.

After amplitude and timing calibration, the interleaved performance improves significantly, achieving an SFDR and SIRR of 76.2 $dB$, as well as an SNDR of 59.5 $dB$ for a 0.997 $GHz$ output. The SNDR performance correlates very well with the simulation results of Fig. 5.8, indicating that random jitter from the clock buffering is the dominating factor. Including $\approx 7 \, dB$ of measurement loss, an output power of $-7.6 \, dBm$
Figure 5.15: Measured second order HP output spectrum across an 800 MHz bandwidth at a sample rate of 2 GS/s. The noise density is labeled at 100 MHz steps and ideal noise shaping is overlaid for reference.

is recorded, closely matching extracted simulation results. Within the same HP bandwidth, Fig. 5.14(b) shows a calibrated IM3 of $-80.3\, \text{dBc}$ from a two tone test with 1 MHz spaced inputs. In Fig. 5.14(c-d), one and two tone tests are also shown for the second BP output. Within a 50 MHz bandwidth (2.5% of $f_S$), the calibrated DAC achieves an SFDR of 65.4 dB and in-band SNDR of 56 dB while measuring an IM3 of $-67.9\, \text{dBc}$. Note that the BP results are slightly degraded compared to the HP mode, due to the large zero-order hold attenuation in the second Nyquist zone. In Fig. 5.15, a wideband plot of the second order HP output is given. Over the 800 MHz bandwidth, the measured noise density is labeled at 100 MHz intervals and compared well to an ideal $\Delta\Sigma$ noise floor.
Figure 5.16: DAC performance across varying frequencies for (a) HP mode and (b) BP mode.

The performance of the DAC across sample rate for second order HP modulation is shown in Fig. 5.16(a). Interleaving achieves a minimum SIRR of 73.5 dB due to accurate amplitude and timing calibration between each DAC. The SFDR performance remains above 65 dB up to an output frequency of 1.5 GHz. The IM3 is measured at a minimum of −80.3 dBc up to 1 GHz and increases to −67 dBc at 1.5 GHz. Figure 5.16(b) shows the measured results of the BP modulation using the second passband, yielding an SFDR above 65 dB and IM3 better than −62 dBc across the frequency band.

In Fig. 5.17, the DAC sample rate is set to 3 GS/s, utilizing both the HP and BP output bands to synthesize 5 MHz WCDMA signals at 1.5 GHz and 2.25 GHz, respectively. The DAC achieves an ACLR of −65.4 dBc in the HP mode, and −59.3 dBc in the BP mode. Figure 5.16(a-b) plots the DAC WCDMA ACLR performance across sample rate for a 5 MHz bandwidth in both the HP and BP passbands. In Fig. 5.16(b), a sample rate of 2.6 GS/s is used to create a BP 1.95 GHz WCDMA output achieving an ACLR of −60.2 dBc. Figure 5.18 shows the generation of a 20 MHz
Figure 5.17: ACLR of a 5 MHz WCDMA signal at 3 GS/s in the (a) 1.5 GHz HP band and (b) 2.25 GHz BP band.

LTE signal at 1 GHz (HP, 2 GS/s) and 1.95 GHz (BP, 2.6 GS/s), yielding an ACLR of $-66.4 \, dBc$ and $-59.2 \, dBc$, respectively.

In Table 5.1, the power consumption of the primary functional blocks is given at 2 GS/s along with the respective layout area. The 9-stage low jitter CML buffers consume 342 mW with the layout distributed across chip while the two pipelined DSMs use a combined 218 mW. With technology scaling beyond 0.13 $\mu$m, the power
Figure 5.18: ACLR of a 20 MHz LTE bandwidth in the (a) 1 GHz HP band and (b) 1.95 GHz BP band.

Consumption of the DSMs and drivers is expected to drop considerably due to both lower supply voltage and reduced digital loading. The low-distortion DAC current cells consume 53 $mW$ from a 3.3 $V$ supply while the retiming path consumes 174 $mW$ divided between a 1.5 $V$ digital supply and a 2 $V$ driver supply. The input LVDS consumes a total of 56 $mW$ for all fourteen input receivers.
Table 5.1: Power Consumption at 2 GS/s and Active Area Breakdown

<table>
<thead>
<tr>
<th>Block</th>
<th>Pwr. @ 2 GS/s [mW]</th>
<th>Area [mm^2]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Current Cells (3.3 V)</td>
<td>53</td>
<td>0.255</td>
</tr>
<tr>
<td>Retiming Data Path</td>
<td>174</td>
<td>0.130</td>
</tr>
<tr>
<td>DSM (1.5 V)</td>
<td>109 (x2)</td>
<td>0.360 (x2)</td>
</tr>
<tr>
<td>CML Clocking (2.3 V)</td>
<td>342</td>
<td></td>
</tr>
<tr>
<td>LVDS RX (1.8 V)</td>
<td>4 (x14)</td>
<td>0.0115 (x14)</td>
</tr>
<tr>
<td>Total Chip</td>
<td>843</td>
<td>15</td>
</tr>
</tbody>
</table>

5.2.3 Comparison

Table 5.2 compares the interleaved ΔΣ DAC performance at 2 GS/s and 3 GS/s with other ΔΣ mixing-DACs and transmitters. Among the designs, the proposed architecture uniquely enables the use of HP ΔΣ modulation by suppressing the first DAC image below −76 dBc. For sample rates above 1 GS/s and bandwidths above 15 MHz, the proposed DAC exhibits the highest SFDR, IM3, and SNDR among previously reported ΔΣ DACs. Due to its excellent linearity, the highest WCDMA ACLR is achieved, while first reporting an LTE ACLR of −66.4 dBc for a 1 GHz output.

Table 5.2: ΔΣ RF-DAC Measurement Comparison

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>[19]</td>
<td>180 CMOS</td>
<td>Mix.</td>
<td>LP</td>
<td>0.514</td>
<td>0.942</td>
<td>-14.65</td>
<td>75</td>
<td>-70.8</td>
<td>53 ^ 1</td>
<td>17.5</td>
<td>-/-</td>
<td>36</td>
</tr>
<tr>
<td>[20]</td>
<td>130 CMOS</td>
<td>Mix.</td>
<td>LP</td>
<td>2.625</td>
<td>5.25</td>
<td>-8</td>
<td>52</td>
<td>-</td>
<td>49</td>
<td>200</td>
<td>-/-</td>
<td>10</td>
</tr>
<tr>
<td>[21]</td>
<td>250 CMOS</td>
<td>Mix.</td>
<td>BP</td>
<td>0.25</td>
<td>1.062</td>
<td>-5.4</td>
<td>75</td>
<td>-64.7</td>
<td>72 ^ 1</td>
<td>15</td>
<td>-/-</td>
<td>12.75</td>
</tr>
<tr>
<td>[26]</td>
<td>90 CMOS</td>
<td>Direct</td>
<td>DP</td>
<td>4.0</td>
<td>1.0</td>
<td>3.1</td>
<td>-</td>
<td>-</td>
<td>53.5</td>
<td>50</td>
<td>-53.6/-</td>
<td>71</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2.6</td>
<td>1.95</td>
<td>-8.6</td>
<td>-</td>
<td>-</td>
<td>46.5</td>
<td>30</td>
<td>-44.3/-</td>
<td>39</td>
</tr>
</tbody>
</table>

Rev 1 180 BiCMOS Direct HP 2.0 1.0 -7.5^* 58.5 52.5 54.5^ 50 -62/-59.8 55

Rev 2 180 BiCMOS Direct HP 2.0 1.0 -0.6^* 76.2 -80.0 55.5 100 -67.2/-66.4 53
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>[98]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

^1 2nd Nyquist Zone  ^1 SNR  ^1 Testing losses removed
In contrast with the first $\Delta\Sigma$ DAC revision, the second design displays across-the-board performance gains (with the exception of data path power consumption owing to the CML topology in revision 2). The improved SNR is a direct result of the low jitter clock and data paths, the reduced IM3 is due to improved matching from calibration, and the SFDR increase is a result of low data dependency of the output timing. Overall, the interleaved $\Delta\Sigma$ architecture is validated by the two design revisions which show predictable RF performance allowing for incremental design improvement. With increased scaling, the topology will see commensurate improvements in speed, power, and RF performance.
Chapter 6: Conclusion and Future Work

6.1 Work Summary and Conclusion

This work describes the theory, design, application, and testing of an interleaved, multi-mode ΔΣ DAC, developed as a means of direct digital-to-RF synthesis for highly configurable transmitters. The design demonstrates reconfigurable RF synthesis via a high-speed, low-resolution DAC fed with ΔΣ modulated data. Utilizing differential parallel data paths, the design cancels the first DAC image, leveraging the 1st and 2nd Nyquist zones for RF output. The interleaved architecture uniquely facilitates the use of HP ΔΣ modulation by canceling the image replica that falls in-band, enabling the wide HP passband for RF output. The DAC can also be configured for BP modulation to enable signal generation in the first or second Nyquist zone.

The implementation of this design is enabled by a novel LVDS pulse receiver. The fully integrated receiver architecture utilizes on-chip AC coupling to remove stringent common-mode standards, facilitate optimum sense-amp biasing, and reduce the area consumed by the AC coupling capacitors. By accurately sizing the input capacitors, the receiver is able to attenuate incoming data, creating return-to-zero pulses which limit ISI jitter without compromising sensitivity. The design implements a latch
within the receiver amplifier using a toggled bias structure which holds long runs of invariant data without decay. In this way, the design removes the need for data encoding, simplifying the high-speed data input of the proposed DAC. To verify its operation, the proposed pulse receiver is tested from 1-10 Gb/s and achieves a BER of < $10^{-13}$ at a peak efficiency of 0.54 mW/Gb/s.

With the means of high-speed data I/O verified, the proposed DAC is implemented in two revisions. The first version of the DAC, integrating only the clock generation and low-resolution DACs, achieves an SIRR of 72 dB, validating the proposed image cancellation. However, the design performs only moderately with respect to SFDR, IM3, and SNDR. To improve upon this performance, the deficiencies of the first design are analyzed and a second revision is implemented.

In the redesigned DAC, per-cell amplitude and timing calibration is used to minimize mismatch and improve IM3; the architecture leverages SiGe HBTs with CML drivers in the DAC core to further maximize the system linearity and improve SFDR (due to reduced data-dependent jitter). A CML clock distribution is also used to reduce random timing jitter, significantly improving the output SNDR. The DAC achieves an interleaved image cancellation above 76 dB with both BP and HP ΔΣ shaping validated. At a sample rate of 2 GS/s, the proposed design achieves an in-band SFDR of 76.2 dB, IM3 of -80 dBc, and SNDR of 59.5 dB over a 100 MHz bandwidth. The high spectral purity allows the DAC to generate WCDMA and LTE signals with an ACLR of -67.2 dBc and -66.4 dBc, respectively, at 1 GHz. Compared to recently reported ΔΣ DACs, the results show the highest linearity performance when using sample rates greater than 1 GHz, and bandwidths greater than 15 MHz.
6.2 Future Work

The interleaved, multi-mode $\Delta\Sigma$ RF-DAC described in this work promises expanded bandwidth and greater flexibility over current $\Delta\Sigma$ RF-DAC designs. As scaling continues to reduce power consumption and improve on-chip digital processing, such all-digital methods of RF synthesis will become a crucial part of SDR transmitters. Critical to the development and integration of this work into larger wireless systems are two broad categories of future research: the improvement of the current design with advanced process technology and the optimization of the design for system integration.

6.2.1 Scaling Improvements

While the proposed design achieves state-of-the-art performance for a direct digital-to-RF architecture, much can be gained from its implementation in a more advanced process technology. As is commonly known, CMOS technology scaling has two great advantages - the reduction of power consumption via reduced supply voltage (and smaller circuit loading) and the increased processing speed due to decreased parasitic capacitances. If the topology were designed into such processes, it would see significant improvements in system performance.

First, the reduced power consumption of a 65 nm or 45 nm technology would greatly reduce the $\sim 800 \text{mW}$ of power consumption, especially in the high-speed DSM, the low-jitter CML clock distribution, and the CML data path. This improvement would make the architecture more suitable for low-power mobile applications. Second, the reduced capacitive loading of an advanced process would allow the integrated DSM to either 1) dramatically increase its sample rate or 2) implement
complex filter functions capable of more passband frequencies. Currently, the design is limited to 3 notch frequencies while operating up to $3GS/s$. Such advancement would go a long way in boosting the flexibility and performance of the proposed architecture. While highly-linear SiGe HBTs are not currently available in the cited process nodes, circuit techniques such as return-to-zero outputs can be used to reduce the non-linear effects of standard CMOS devices.

6.2.2 System Integration

Further afield, for the proposed RF-DAC to be successfully integrated into a high-functioning wireless system, several key steps need to be taken:

- DAC Resolution Trade-Off Analysis
- High-Power (or III-V) Implementation
- High-Speed Digital Mixing
- Reconfigurable Filter Design
Resolution Analysis

In the preceding work, a 3-bit design has been assumed for both its simple implementation and also its challenges (non-linearity and matching requirements) which require significant design work to overcome. However, by leaving the DAC resolution static, many design trade-offs such as in-band/out-of-band SNR and reconstruction filter requirements were ignored. An analysis of this trade space, while design specific, would be necessary for future development.

High-Power Design

The ultimate goal of many digital transmitters is the complete removal of all analog/RF signal conditioning, including the power amplifier. However, as discussed in this work, the efficiency of this architecture does not lend itself to high power (watt level) outputs. Instead, the end goal of this design would be to drive a high-power PA without the need for any buffering. To accomplish this, the power of the ∆Σ modulated output needs to be increased from $0\, dBm$ to around $-15\, dBm$. This can be done in one of several ways, but the most ideal would be a III-V implementation of the RF-DAC. Given the unique low-resolution topology of the design, it is uniquely suited for a high-power process technology which are typically susceptible to high mismatch and poor integration levels. Key to this implementation would be a high speed CMOS to III-V digital interface capable of transporting the ∆Σ modulated data across the voltage domain barrier.

Mixing

Assumed in the RF-DAC design is an up-converted data source preceding the DSM, pictured in Fig. 6.2. This operation, typically implemented with a clocked
LO and an AND gate, takes a baseband signal up to RF frequencies. For system level integration, such a block would be implemented on-chip before the DSM. The practicalities of such up-conversion, however, have not yet been explored for the proposed design.

Figure 6.2: Future work for system integration

Reconfigurable Filtering

Finally, and probably most importantly, the post-DAC reconstruction filtering must be addressed before system integration. Currently, the in-band performance of the RF-DAC meets the noise floor, spectrum mask, and spurious tone requirements of many wireless systems, but the out-of-band noise does not. This high noise is a result of the DSM shaping and would corrupt surrounding transmissions if left unattenuated. The challenge of this filter lies in its need for reconfigurable operation matching that of the DSM feedback filter. Several possibilities for the filter design exist (e.g. N-path filtering, semi-digital filtering, reconfigurable LC filtering), but such a design has yet to be implemented for the proposed RF-DAC.
6.3 Final Thoughts

Altogether, this work gives only a glimpse into the potential of direct digital-to-RF synthesis as a means of reconfigurable transmitter design. With future work, and as digital processing becomes simultaneously cheaper and more powerful, digital radio systems will become increasingly useful for both their adaptability in changing environments and their ease of deployment. Though we may look beyond specifications and speculate about the future capabilities of such systems, their true power will only be comprehended in hind-sight. Through the presented paradigm or others, highly digital architectures will continue to revolutionize the capability and ubiquity of wireless communications.
Appendix A: ∆Σ DAC Model

This appendix contains the Matlab/Simulink model used to test the ideal performance of the interleaved, ∆Σ DAC as well as the DAC performance under amplitude, phase, and current cell mismatch. The DSM output is calculated by a Simulink model, given in Fig. A.1, and the mismatch and channel summation are subsequently accomplished in code.

Matlab Script:

```matlab
1 clear all; % reset everything
2
3 %% Simulation Variables
4
5 nmod = (2^18)/2^3; % number of samples
6 kmod = 1024; % number of startup samples to remove
7 acmag = 5; % Input AC mag (DC = 5, Max Peak <= 10)
8 tQ = 25e-11; % Simulink sample rate (2*fs of singlel DAC)
9 Bin_Num = 8135*8+5; % Frequency of signal by bin: 8022*8+1
10 % 8033*8+5 8042*8+1 8135*8+5 8072*8+7
11 % 8072+9 8022+1 7967+7 8135+7
12 % BP: 4020+1 12020+1 12050+1 12210+1
13 M = 3; % Number of output bits DS
14 T = 15; % Number of truncated bits DS
15 Mode = 0; % Mode = 0 HP - Mode = 1 BP
16 delta = .001; % ILSB [A]
17 R = 50; % Single Ended Termination [Ohms]
18 Offset = 2^(M+T-1)+2^(M+T-2); % DS static offset
19
20 % Calculated Signal Power
21 P_Signal = ((2^(M+T-1)-1)/(2^T)*delta/2^.5*acmag/5)^2*R*2/.001;
22
23 if Mode == 0
24     delay = 2;
25 ```

107
```plaintext
elseif Mode == 1
    delay = 4;
end

fmod = Bin_Num / (nmod*tQ);   % Input Frequency

%% Mismatch Control Variables
PhaseMismatch_toggle = 0;     % [0 = off; 1 = on]
GainMismatch_toggle = 0;      % [0 = off; 1 = on]

if GainMismatch_toggle == 1
    % Overall Gain of DAC0
    G0 = 1.001;
    % Cell by cell gain of DAC0 (size must be equal to DAC
    % resolution due to assumption of fully unary architecture)
    %G0SYS = [1 1 1 1 1 1 1];
    G0SYS = [1 1.0012 .9966 .9977 .9897 .9977 1.0035];
    % Could add random Num function here for random mismatch
    G0RND = [1 1 1 1 1 1 1];

    % Overall Gain of DAC1
    G1 = 1;
    % Cell by cell gain of DAC1 (size must be equal to DAC
    % resolution due to assumption of fully unary architecture)
    %G1SYS = [1 1 1 1 1 1 1];
    G1SYS = [1 1.0046 .9931 .9874 1.0023 1.0006 1.009];
    % Could add random Num function here for random mismatch
    G1RND = [1 1 1 1 1 1 1];
endif

if PhaseMismatch_toggle == 1
    % Amount of oversampling determines resolution of phase error
    % High oversampling values consume large amounts of memory
    Oversample = 10;
    % Systematic DAC0 phase error
    PHI0 = 1;
    % Per cell phase error
    PHI0SYS = [0 0 0 0 0 0 0];
    % Random function could be added here
```

PHI0RND = [0 0 0 0 0 0 0];

% Systematic DAC1 phase error
PHI1 = 0;

% Per cell phase error
PHI1SYS = [0 0 0 0 0 0 0];

% Random function could be added here
PHI1RND = [0 0 0 0 0 0 0];

else
    % No oversampling needed if there is no phase error
    Oversample = 1;
end

% Run Simulink
sim('DS_DAC_HP_Interleaved_Model3_Nbit');

% This model does not check for overflow and handles it ideally by
% expanding the value of the output past the resolution of the DAC
%(+ or -)

% Adding Mismatch
% Systematic and Cell Timing Error - Waveform adjusted for error
if PhaseMismatch_toggle == 1
    HP_DS_RFDAC0P = zeros(1,Oversample*length(HP_DS_RFDAC0));
    HP_DS_RFDAC1P = zeros(1,Oversample*length(HP_DS_RFDAC1));

    for x = 1:length(HP_DS_RFDAC0)
        for y = 1:Oversample
            for z = 1:2^M-1
                % First DAC
                if HP_DS_RFDAC0(x) >= z*delta && ((x-1)*
                    Oversample + y+PHI0+PHI0SYS(z)+PHI0RND(z)) ...
                    <= (Oversample*length(HP_DS_RFDAC0))
                    HP_DS_RFDAC0P((x-1)*Oversample + ...
                    y+PHI0+PHI0SYS(z) + PHI0RND(z)) = delta + ...
                    HP_DS_RFDAC0P((x-1)*Oversample + ...
                    y+PHI0+PHI0SYS(z)+PHI0RND(z));
            end
        end
    end

    % Second DAC
    if HP_DS_RFDAC1(x) >= z*delta && ((x-1)*
        Oversample + y+PHI1+PHI1SYS(z)+ ...
        PHI1RND(z)) <= (Oversample* ...
        length(HP_DS_RFDAC1))
HP_DS_RFDAC1P((x-1)*Oversample + y+PHI1+PHI1SYS(z) + PHI1RND(z)) =
   delta + HP_DS_RFDAC1P((x-1)*Oversample + y+PHI1+PHI1SYS(z)+PHI1RND(z));

end
end
end

HP_DS_RFDAC0 = HP_DS_RFDAC0P;
HP_DS_RFDAC1 = HP_DS_RFDAC1P;
clear HP_DS_RFDAC1P HP_DS_RFDAC0P

% Systematic and Cell Gain Error - Waveform adjusted for error
if GainMismatch_toggle == 1
   for x = 1:length(HP_DS_RFDAC0)
      for z = 1:2^M-1
         if HP_DS_RFDAC0(x) == z*delta
            HP_DS_RFDAC0(x) = delta * sum(G0SYS(1:z).*G0RND(1:z))*G0;
         end
      end
      if HP_DS_RFDAC1(x) == z*delta
         HP_DS_RFDAC1(x) = delta * sum(G1SYS(1:z).*G1RND(1:z))*G1;
      end
   end
end

%% Computing PSD
HP_DS_RFDAC = 2*R.*(HP_DS_RFDAC0(1:Oversample* 
   ((nmod+kmod)*2)+1) + HP_DS_RFDAC1(1:Oversample* 
   ((nmod+kmod)*2)+1));
RBW = (.5/tQ)/(nmod/2); % Desired plotting RBW
last_HP_DS_RFDAC = HP_DS_RFDAC(length(HP_DS_RFDAC)-nmod* 
   Oversample+1:length(HP_DS_RFDAC));

% FFT Calculation
FFT_HP_DS_RFDAC = fft(last_HP_DS_RFDAC, nmod*Oversample);

% PSD Calculation
FFT_V = FFT_HP_DS_RFDAC(1:nmod*Oversample/2+1);
PSD = abs(FFT(V)/(nmod*Oversample)).^2./((2*R*.001));

% PSD = (tQ/Oversample)/(nmod*Oversample) * ...
% abs(FFT(V)).^2./((2*R*.001));

PSD(2:end-1) = 2*PSD(2:end-1);

% To dB
dB_Pout = 10*log10(PSD);

% Frequency - X Axis
F = (0:nmod*Oversample/2) / (nmod*tQ);

% Analytical Output - can be compared with Simulated Spectrum
Erms = delta*R*4/(12)ˆ.5*(tQ)ˆ.5;

NTF_HP_2 = (4) .* (exp(-2i.*pi.*F.*tQ)) .* cos(2*pi.*F.*tQ) .* ...
          cos(2*pi.*F.*tQ)*Erms;
NTF_HP_2(Bin_Num+1) = (P_Signal).^5;
NTF_HP_2 = NTF_HP_2.*sin(2*pi.*F.*tQ)./(2*pi.*F.*tQ);

Power_HP_2 = (abs(NTF_HP_2)).^2./((R*.001)*RBW);
Power_HP_2(Bin_Num+1) = Power_HP_2(Bin_Num+1)*((R*.001)/RBW);

% SNR Calculation: assumes signal is within bandwidth
SNR_BW = 50e6; % SNR Bandwidth for Calculation
BinWidth = 1/(nmod*tQ);
CenterF = max(F)/2;
LowF = CenterF - SNR_BW/2;
HighF = CenterF + SNR_BW/2;

IndexLow = find(F >= LowF,1);
IndexHigh = find(F >= HighF,1);

Noise_P_Calc = sum(Power_HP_2(IndexLow:IndexHigh)./...
          RBW.*BinWidth) - Power_HP_2(Bin_Num+1)./RBW.*BinWidth;
Signal_P_Calc = Power_HP_2(Bin_Num+1);
SNR_Calc = 10*log10(Signal_P_Calc)-10*log10(Noise_P_Calc);

Image_P_Sim = 0;
if PhaseMismatch_toggle == 1 || GainMismatch_toggle == 1
    Image_P_Sim = PSD(nmod/2-Bin_Num+1);
end

Noise_P_Sim = sum(abs(PSD(IndexLow:IndexHigh))./RBW.*BinWidth);
Signal_P_Sim = PSD(Bin_Num+1);
SNR_Sim = 10*log10(Signal_P_Sim)-10*log10(Noise_P_Sim - ...
          (Signal_P_Sim+Image_P_Sim)*BinWidth);
%% Plotting Result

% Plot First two Nyquist zones
figure('Color',[1.0 1.0 1.0],'Position',[1 1 1111 599])
plot(F/1e9,dB_Pout,'LineWidth',3,'Color', [200/255 0 0])
hold on
plot(F/1e9,10*log10(Power_HP_2),'LineWidth',3,'Color', [0 0 0])
grid on;
set(gca, 'Box', 'on', 'Linewidth', 2,'fontname','Arial','fontsize',32);
set(gca,'Color',[.941 .941 .941])
set(gca,'XLim',[0,1/tQ/2e9])
set(gca,'YLim',[-150,10])
xlabel('Frequency [GHz]')
ylabel('Power [dBm]')
title('RF-DAC Spectrum')
h = legend('Simulated: Non-Ideal','Analytic: Ideal',...
     'Location','SouthWest');

% Plot close in Spectrum
figure('Color',[1.0 1.0 1.0],'Position',[1 1 1111 599])
plot(F/1e9,dB_Pout,'LineWidth',3,'Color', [200/255 0 0])
hold on
plot(F/1e9,10*log10(Power_HP_2),'LineWidth',3,'Color', [0 0 0])
grid on;
set(gca, 'Box', 'on', 'Linewidth', 2,'fontname','Arial','fontsize',32);
set(gca,'Color',[.941 .941 .941])
if Mode == 0
   set(gca,'XLim',[0.45/tQ/2e9,0.55/tQ/2e9])
elseif Mode == 1
   set(gca,'XLim',[0.2/tQ/2e9,0.3/tQ/2e9])
end
set(gca,'YLim',[-160,20])
xlabel('Frequency [GHz]')
ylabel('Power [dBm]')
title('RF-DAC Spectrum')

% Calculate Spectrum Characteristics
OutputPower = max(dB_Pout(2:end));
NF = 10.*log10(mean(10.^(dB_Pout(Bin_Num+3:Bin_Num+103)./10))) ...
    - 10*log10(RBW);
SIRR = dB_Pout(Bin_Num+1) - dB_Pout(2*(nmod/4+1-Bin_Num-1))...
SFDR.3rd = dB_Pout(Bin_Num+1) - dB_Pout(4*(nmod/4+1-Bin_Num-1)... +Bin_Num+1);

% Display values

text(0.45/tQ/2e9+.001,-10,['P_{OUT} = ' num2str(round(OutputPower*10)/10) ' dBm'], 'FontSize', 14, ... 'BackgroundColor', [1 1 1]);
text(0.45/tQ/2e9+.001,-25,['NF = ' num2str(round(NF*10)/10) ' dBm/Hz'], 'FontSize', 14, 'BackgroundColor', [1 1 1]);
text(0.45/tQ/2e9+.001,-40,['SIRR = ' num2str(round(SIRR*10)/10) ' dB'], 'FontSize', 14, 'BackgroundColor', [1 1 1]);
text(0.45/tQ/2e9+.001,-55,['SFDR.3rd = ' num2str(round(SFDR.3rd*10)/10) ' dB'], 'FontSize', 14, ... 'BackgroundColor', [1 1 1]);

h = legend('Simulated: Non-Ideal','Analytic: Ideal',... 'Location','NorthEast');
Figure A.1: Simulink model of the multi-mode, interleaved $\Delta\Sigma$ DAC.
Appendix B: $\Delta \Sigma$ DAC Random Jitter Model

This appendix contains the Matlab model used to test the SNR performance of the interleaved, $\Delta \Sigma$ DAC with random data jitter. The DSM output is computed, oversampled, and then random jitter is applied systematically to each DAC or individually to each DAC cell.

Matlab Script:

```matlab
function [SNR_Ideal, SNR_AllJitter, SNR_DACJitter, ...]
    SNR_CellJitter] = DS_2ndOrderHP_Jitter1(...
    Num_Samples, R0, ILSB, Resolution, Bits,...
    Fs, Bin, SNR_BW, OSR, Per_Cell,...
    Cell_Systematic_Timing_Error1, ...
    Cell_Systematic_Timing_Error2, ...
    Cell_Systematic_Amplitude_Error1, ...
    Cell_Systematic_Amplitude_Error2, Sigma, Plot)

% Simulation of 2nd Order, High Pass DS DAC with Clock Jitter
% Must be run by Run_DS_2ndOrderHP_Jitter1.m
% Can put an % before function to use as regular .m script
% Created by: Jamin McCue

%% Beginning Calculations

% Frequency of Sine wave
Fsine = Fs*Bin/Num_Samples;

% Samples per Sine Wave
% # of Samples / Samples per Cycle = Integer
% ie. (80*512*2 - 1) / 4.0959 = 10000
Samples_per_Cycle = Fs/Fsine;

%% Interleaved Signal Generation

% Starting Phase
Phase = 0;
```

115
% Signal Backoff for DS and setting of DC
if Bits == 4
    Signal_Amplitude = .74; % N = 4
    Signal_DC = (2^Resolution-1)/2 + 2^(Resolution-4);
elseif Bits == 3
    Signal_Amplitude = .62; % N = 3
    Signal_DC = (2^Resolution-1)/2 + 2^(Resolution-3) + ...
                2^(Resolution-4);
elseif Bits == 2
    Signal_Amplitude = .25; % N = 2
    Signal_DC = (2^Resolution-1)/2 + 2^(Resolution-2) + ...
                2^(Resolution-4) + 2^(Resolution-5) + 2^(Resolution-6);
end

% Phase Increment of Sine Wave
Phase_inc = 2*pi/Samples_per_Cycle;
% Initialize Variable
Output = zeros(1,Num_Samples);
% Initial Input: Sine Wave
for x = 1:length(Output)
    Output(x) = round(Signal_Amplitude*(2^Resolution-1)/2* ...
                      sin(Phase + (x-1)*Phase_inc)+Signal_DC);
end

% Error bits are Fractional
Output = Output / 2^(Resolution - Bits);
% Creation of the two channel data
Output1 = Output(1:2:end);
Output2 = Output(2:2:end);

%% Plot Sine Wave
figcnt = 1;
if Plot
    hfig = figure(figcnt); figcnt = figcnt+1;
    set(gcf, 'PaperUnits', 'inches', ...
        'PaperPosition', [0 0 4 3], ...
        'PaperSize', [4 3], ...
        'Color', [1 1 1], ...
        'Position', [1 1 650 500]);
    plot((0:1/Fs:1/Fs*length(Output)-1/Fs).*1e9,Output, '-k', ...
         'Linewidth', 4, 'MarkerEdgeColor', 'k', ...
         'MarkerFaceColor', 'w', 'MarkerSize', 8);
    hold on
    plot((0:2/Fs:2/Fs*length(Output1)-1/Fs).*1e9,Output1,'o', ...
         'Linewidth', 4, 'MarkerEdgeColor', [0 0 200/255], ...
         'MarkerFaceColor', 'w', 'MarkerSize', 8);
    plot((1/Fs:2/Fs:2/Fs*length(Output2)-1/Fs).*1e9,Output2,'o', ...
         'Linewidth', 4, 'MarkerEdgeColor', [200/255 0 0], ...
         'MarkerFaceColor', 'w', 'MarkerSize', 8);
set(gca, 'fontsize', 18, 'fontname', 'Arial');
set(gca, 'Box', 'on', 'Linewidth', 3);
xlabel('Time [ns]'); ylabel('[V]');
set(gca, 'XTick', [0:50:2^10/Fs*1e9]); set(gca, 'YTick', [0:1:2^Bits+1]);
axis([0 2^10/Fs*1e9 0 2^Bits+1]);
grid on;
h = legend('Input Sine','Data: Chan1','Data: Chan2', ...
'Location','NorthEast');
set(h, 'LineWidth', 2)
end

%% DS Modulation

% DC offset of DS Modulator input
Offset = 0;

% Initializing feedback memory
Quant1a = zeros(1, length(Output)/2);
Quant1b = zeros(1, length(Output)/2);
Quant2a = zeros(1, length(Output)/2);
Quant2b = zeros(1, length(Output)/2);

% Initializing DS Outputs
DS_Output1 = zeros(1, length(Output)/2);
DS_Output2 = zeros(1, length(Output)/2);

% Initializing Sum/Diff Outputs
Sum1 = zeros(1, length(Output)/2);
Sum2 = zeros(1, length(Output)/2);

% Implementation of DS Modulation: 2nd Order HP
for x = 1:length(Output1)
  % Output of Sum/Diff
  Sum1(x) = Output1(x) - 2*Quant1a(x) - Quant1b(x) + Offset;
  Sum2(x) = Output2(x) - 2*Quant2a(x) - Quant2b(x) + Offset;

  % MSBs (Bits after truncation)
  DS_Output1(x) = fix(Output1(x) - 2*Quant1a(x) - Quant1b(x) + Offset);
  DS_Output2(x) = fix(Output2(x) - 2*Quant2a(x) - Quant2b(x) + Offset);

  % Loading of feedback memory with Truncated LSBs
  if x ~= length(Output1)-1
    Quant1a(x+1) = Sum1(x) - DS_Output1(x);
    Quant1b(x+1) = Quant1a(x);
    Quant2a(x+1) = Sum2(x) - DS_Output2(x);
    Quant2b(x+1) = Quant2a(x);
  end
end
% Repeat Bits so that signals have proper phase
% alignment during addition
Output1(1:2:length(Output)) = DS_Output1(1:end);
Output1(2:2:length(Output)) = DS_Output1(1:end);
Output2(2:2:length(Output)) = DS_Output2(1:end);
Output2(3:2:length(Output)) = DS_Output2(1:end-1);
Output2(1) = DS_Output2(end);

% If Per_Cell is on, determine output of each DAC cell
if Per_Cell
    Output_Bit1 = zeros(2^Bits-1,length(Output1));
    Output_Bit2 = zeros(2^Bits-1,length(Output2));
    for x = 1:length(Output1)
        if Output1(x) > 0
            Output_Bit1(1:Output1(x),x) = 1;
        end
        if Output2(x) > 0
            Output_Bit2(1:Output2(x),x) = 1;
        end
    end
end

% Check for Overflow
if max(Output1) > 2^Bits-1 || max(Output2) > 2^Bits-1
    disp('Error: Delta Sigma Overflow Occured - Output')
end
if min(Quant1a) < 0 || min(Quant1b) < 0 ... || min(Quant2a) < 0 || min(Quant2b) < 0
    disp('Error: Delta Sigma Overflow Occured - Error Feedback')
end

% Addition of the two channels before jitter
DS_Output = Output1 + Output2;

%% Plot DS Output
if Plot
    hfig = figure(figcnt); figcnt = figcnt+1;
    set(gcf, 'PaperUnits', 'inches', ... 'PaperPosition', [0 0 4 3], ...
         'PaperSize', [4 3], ...
         'Color', [1 1 1], ...
         'Position', [1 1 650 500]);
    plot((0:1/Fs:1/Fs*length(DS_Output)-1/Fs).*1e9,DS_Output,... '-k', 'Linewidth',4, 'MarkerEdgeColor', 'k', ...
         'MarkerFaceColor', 'w', 'MarkerSize', 8);
hold on
plot((0:2/Fs:2/Fs*length(Output1)-1/Fs).*1e9,Output1,'-','Color',[0 0 200/255],'Linewidth', 4);
plot((1/Fs:2/Fs:2/Fs*length(Output2)).*1e9,Output2,'-','Color',[200/255 0 0],'Linewidth', 4);
set(gca,'fontsize', 18,'fontname','Arial');
set(gca, 'Box', 'on', 'Linewidth', 3);
xlabel('Time [ns]'); ylabel('X \cdot I_{LBS} [A]');
set(gca, 'XTick', [0:50:2^10/Fs*1e9]);
set(gca, 'YTick', [0:2:2^(Bits+1)-1]);
axis([0 2^10/Fs*1e9 0 2^(Bits+1)-1]);
grid on;
h = legend('Interleaved Output','Channel1 Output',...
    'Channel2 Output', 'Location','NorthEast');
set(h, 'LineWidth', 2)
end

%% Plot Ideal DS Output: Frequency Domain

[psdestx1,Fxx1] = periodogram(Output1*2*R0*ILSB,...
    hanning(length(Output1)), length(Output1),Fs/1e9,'power');

[psdestx2,Fxx2] = periodogram(Output2*2*R0*ILSB,...
    hanning(length(Output2)), length(Output2),Fs/1e9,'power');

[psdestx,Fxx] = periodogram(DS_Output*2*R0*ILSB,...
    hanning(length(DS_Output)), length(DS_Output),Fs/1e9,'power');

if Plot
    fig = figure(figcnt); figcnt = figcnt+1;
    set(gcf, 'PaperUnits', 'inches',...
        'PaperPosition', [0 0 4 3],...
        'PaperSize', [4 3],...
        'Color', [1 1 1],...
        'Position',[1 1 750 500]);
    plot(Fxx,10*log10(psdestx), '-k',...
        'Linewidth',4, 'MarkerEdgeColor', 'k',...
        'MarkerFaceColor', 'w', 'MarkerSize', 8);
    hold on
    plot(Fxx1,10*log10(psdestx1),'-',...
        'Color',[0 0 200/255],'Linewidth', 4);
    plot(Fxx2,10*log10(psdestx2),'-',...
        'Color',[200/255 0 0],'Linewidth', 4);
    set(gca,'fontsize', 18,'fontname','Arial');
    set(gca, 'Box', 'on', 'Linewidth', 3);
xlabel('Frequency [GHz]'); ylabel('Power [dBm]');
    set(gca, 'XTick', [0:.25:max(Fxx)]); set(gca, 'YTick',...
        [-150:25:50]);
axis([0 max(Fxx) -125 25]);
grid on;
h = legend('Interleaved Output','Channel1 Output',...
'Channel2 Output', 'Location','NorthEast');
set(h, 'LineWidth', 2)
end

%% Calculate Ideal SNR

% Lower Frequency Limit
FL = Fs/4 - Fs/4 * SNR_BW/2;
% Upper Frequency Limit
FU = Fs/4 + Fs/4 * SNR_BW/2;
% FFT bin for low end
BinL = floor(F_L * Num_Samples / Fs) + 1;
% FFT bin for high end
BinU = ceil(F_U * Num_Samples / Fs) + 1;
% Total Power in SNR BW
BW_Pwr_Ideal = sum(psdestx(BinL:BinU));
% Signal Power in SNR BW
Signal_Pwr_Ideal = sum(psdestx(Bin:Bin+2));
% Noise Power in SNR BW
Noise_Pwr_Ideal = BW_Pwr_Ideal - Signal_Pwr_Ideal;
% SNR
SNR_Ideal = 10*log10(Signal_Pwr_Ideal/Noise_Pwr_Ideal);

if Plot
% Plot Markers showing limit of SNR BW
plot(Fxx(BinU),10*log10(psdestx(BinU)), 'o', ...
'Linewidth',4, 'MarkerEdgeColor', [200/255 0 0], ...
'MarkerFaceColor', 'w', 'MarkerSize', 8);
plot(Fxx(Bin_L),10*log10(psdestx(Bin_L)), 'o', ...
'Linewidth',4, 'MarkerEdgeColor', [200/255 0 0], ...
'MarkerFaceColor', 'w', 'MarkerSize', 8);
end

clear psdestx psdestx2 Fxx2 psdestx1 Fxx1 FL FU
clear BW_Pwr_Ideal Signal_Pwr_Ideal Noise_Pwr_Ideal

%% Oversampling

% Initializing Oversampled Outputs
OS_DS_Output = zeros(1,length(DS_Output)*OSR);
OS_DS_Output1 = zeros(1,length(Output1)*OSR);
OS_DS_Output2 = zeros(1,length(Output2)*OSR);

% Vectors containing the normally distributed jitter
% Rounded to integers for use as indices
Jitter = round(normrnd(0, Sigma*(Fs*OSR),1,length(DS_Output)));
Jitter1 = round(normrnd(0, Sigma*(Fs*OSR),1,length(Output1)));
Jitter2 = round(normrnd(0, Sigma*(Fs*OSR),1,length(Output2)));
% Prevent negative index
Jitter(1) = 0;
Jitter1(1) = 0;
Jitter2(1) = 0;

% For Individual Cell Analysis
if Per_Cell
    OS_DS_Output(Bit1) = zeros(2^Bits-1,length(Output(Bit1))*OSR);
    OS_DS_Output(Bit2) = zeros(2^Bits-1,length(Output(Bit2))*OSR);
    Jitter(Bit1) = round(normrnd(0, Sigma*(Fs*OSR),2^Bits-1,...
        length(Output(Bit1)));
    Jitter(Bit2) = round(normrnd(0, Sigma*(Fs*OSR),2^Bits-1,...
        length(Output(Bit2)));

    % Add in timing error
    Jitter(Bit1) = bsxfun(@plus,Jitter(Bit1),...
        Cell_Systematic_Timing_Error1');
    Jitter(Bit2) = bsxfun(@plus,Jitter(Bit2),...
        Cell_Systematic_Timing_Error2');
    Jitter(Bit1(:,1) = 0;
    Jitter(Bit2(:,1) = 0;
end

% Add jitter to data transitions
for x = 2:length(DS_Output)
    if x < length(DS_Output)
        OS_DS_Output(1+OSR*(x-2)+Jitter(x-1):1+OSR*(x-1)... +Jitter(x)) = DS_Output(x);
        OS_DS_Output1(1+OSR*(x-2)+Jitter1(x-1):1+OSR*(x-1)... +Jitter1(x)) = Output1(x);
        OS_DS_Output2(1+OSR*(x-2)+Jitter2(x-1):1+OSR*(x-1)... +Jitter2(x)) = Output2(x);
    end
end

% If Per_Cell is on, populate Jittery Bit Cell data
if Per_Cell
    for y = 1:2^Bits
        OS_DS_Output_BIT1(y,1+OSR*(x-2)+... Jitter(Bit1(y,x-1):1+OSR*(x-1)+... Jitter(Bit1(y,x)) = Output_BIT1(y,x);
        OS_DS_Output_BIT2(y,1+OSR*(x-2)+... Jitter(Bit2(y,x-1):1+OSR*(x-1)+... Jitter(Bit2(y,x)) = Output_BIT2(y,x);
    end
end
% Last Sample must have Jitter of 0 to keep array length equal
elseif x == length(DS.Output)
    OS_DS_Output(1+OSR*(x-2)+Jitter(x-1):1+OSR*(x-1)) = ...
    DS_Output(x);
    OS_DS_Output1(1+OSR*(x-2)+Jitter1(x-1):1+OSR*(x-1)) = ...
    Output1(x);
    OS_DS_Output2(1+OSR*(x-2)+Jitter2(x-1):1+OSR*(x-1)) = ...
    Output2(x);

% Clear variables for memory purposes
clear Jitter1 Jitter2 Output2 Output1 DS_Output Output

% If Per_Cell is on, finish Bit Cell array
if Per_Cell
    for y = 1:2^Bits-1
        OS_DS_Output_Bit1(y,1+OSR*(x-2)+...
            Jitter_Bit1(y,x-1):1+OSR*(x-1))...
            = Output_Bit1(x);
        OS_DS_Output_Bit2(y,1+OSR*(x-2)+...
            Jitter_Bit2(y,x-1):1+OSR*(x-1))...
            = Output_Bit2(x);
    end
% Clear variables for memory purposes
    clear Jitter_Bit1 Jitter_Bit2 Output_Bit2 Output_Bit1
end
end
end

% Multiply in Per Cell Amplitude error
if Per_Cell
    OS_DS_Output_Bit1 = bsxfun(@times,OS_DS_Output_Bit1, ... 
        Cell_Systematic_Amplitude_Error1');
    OS_DS_Output_Bit2 = bsxfun(@times,OS_DS_Output_Bit2, ... 
        Cell_Systematic_Amplitude_Error2');
end

%% Plot Histogram of Jitter
hfig = figure(figcnt); figcnt = figcnt+1;
set(gca, 'LineWidth', 1, 'FontWeight','bold');
set(gcf, 'PaperUnits', 'inches', 
    'PaperPosition', [0 0 4 3], 
    'PaperSize', [4 3], 
    'Color', [1 1 1], 
    'Position',[1 1 750 500]);
% Array Representing Histogram Bars to use
Bars = 1/(Fs.*OSR).*(-100:100).*1e12;
hist(Jitter./(Fs.*OSR).*1e12, Bars)
set(gca, 'FontSize', 18, 'FontName', 'Arial');
set(gca, 'Box', 'on', 'Linewidth', 3);
xlabel('Data Jitter [ps]'); ylabel('Number');
%set(gca, 'XTick', [0:0.25:max(Fxx)]);
%set(gca, 'YTick', [-150:25:50]);
%axis([0 max(Fxx) -125 25]);
grid on;
h = legend('Jitter Dist.', 'Location','NorthEast');
set(h, 'LineWidth', 2)
clear Jitter

%% Plot DS Jitter Output: Frequency Domain

% Frequency Domain of Jittery Outputs
[OS_psdestx1,OS_Fxx1] = periodogram(OS_DS_Output1*2*R0*ILSB,...
    hanning(length(OS_DS_Output1)),length(OS_DS_Output1),...
    OSR*Fs/1e9,'power');

[OS_psdestx2,OS_Fxx2] = periodogram(OS_DS_Output2*2*R0*ILSB,...
    hanning(length(OS_DS_Output2)),length(OS_DS_Output2),...
    OSR*Fs/1e9,'power');

[OS_psdestx,OS_Fxx] = periodogram(OS_DS_Output*2*R0*ILSB,...
    hanning(length(OS_DS_Output)),length(OS_DS_Output),...
    OSR*Fs/1e9,'power');

[OS_psdestx_add,OS_Fxx_add] = ...
    periodogram(OS_DS_Output1.*2.*R0.*ILSB + ... OS_DS_Output2.*2.*R0.*ILSB,...
    hanning(length(OS_DS_Output1)), ... length(OS_DS_Output),OSR*Fs/1e9,'power');

if Per_Cell
    OS_DS_Output.Bit = zeros(1,length(OS_DS_Output.Bit1(1,:)));
    % Add all of the individual bit cells together
    for x = 1:2^Bits-1
        OS_DS_Output.Bit = OS_DS_Output.Bit1(x,:) + ...
        OS_DS_Output.Bit2(x,:) + ...
        OS_DS_Output.Bit;
    end

    [OS_psdestx.Bit,OS_Fxx.Bit] = ...
        periodogram(OS_DS_Output.Bit.*2.*R0.*ILSB,... hanning(length(OS_DS_Output1)), ...
        length(OS_DS_Output),OSR*Fs/1e9,'power');

end

RBW = (OS_Fxx(2)-OS_Fxx(1)).*1e9;
if Plot
    hfig = figure(figcnt); figcnt = figcnt+1;
    set(gcf, 'PaperUnits', 'inches', ...
        'PaperPosition', [0 0 4 3], ...
        'PaperSize', [4 3], ...
        'Color', [1 1 1], ...
        'Position', [1 1 750 500]);
    hold off
    plot(OS_Fxx,10*log10(OS_psdestx), '-k', ...
        'Linewidth', 4, 'MarkerEdgeColor', 'k', ...
        'MarkerFaceColor', 'w', 'MarkerSize', 8);
    hold on
    plot(OS_Fxx1,10*log10(OS_psdestx1), '-', ...
        'Color', [0 0 200/255], 'Linewidth', 4);
    plot(OS_Fxx2,10*log10(OS_psdestx2), '-', ...
        'Color', [200/255 0 0], 'Linewidth', 4);
    plot(OS_Fxx_add,10*log10(OS_psdestx_add), '-', ...
        'Color', [0 200/255 0], ...
        'Linewidth', 4, 'MarkerEdgeColor', 'k', ...
        'MarkerFaceColor', 'w', 'MarkerSize', 8);
    if Per_Cell
        plot(OS_Fxx_Bit,10*log10(OS_psdestx_Bit), '-', ...
            'Color', [1 200/255 0], ...
            'Linewidth', 4, 'MarkerEdgeColor', 'k', ...
            'MarkerFaceColor', 'w', 'MarkerSize', 8);
    end
    set(gca,'fontsize', 18,'fontname','Arial');
    set(gca, 'Box', 'on', 'Linewidth', 3);
    xlabel('Frequency [GHz]'); ylabel('Power [dBm]');
    set(gca, 'XTick', [0:.25:max(Fxx)]);
    set(gca, 'YTick', [-150:25:50]);
    axis([0 max(Fxx) -125 25]);
    grid on;
    h = legend('Interleaved Output: Top Jitter', ...
        'Channel1 Output', 'Channel2 Output', ...
        'Interleaved Output: 2 Channel Jitter', ...
        'Per Bit Jitter', 'Location', 'NorthEast');
    set(h, 'LineWidth', 2)
end

%% SNR Calculations
% Total Power in SNR BW
BW_Pwr_AllJitter = sum(OS_psdestx(Bin_L:Bin_U));
BW_Pwr_DACJitter = sum(OS_psdestx_add(Bin_L:Bin_U));
% Signal Power in SNR BW
Signal_Pwr_AllJitter = sum(OS_psdestx(Bin:Bin+2));
Signal_Pwr_DACJitter = sum(OS_psdestx_add(Bin:Bin+2));

% Noise Power in SNR BW
Noise_Pwr_AllJitter = BW_Pwr_AllJitter - Signal_Pwr_AllJitter;
Noise_Pwr_DACJitter = BW_Pwr_DACJitter - Signal_Pwr_DACJitter;

% SNR
SNR_AllJitter = 10*log10(Signal_Pwr_AllJitter/Noise_Pwr_AllJitter);
SNR_DACJitter = 10*log10(Signal_Pwr_DACJitter/Noise_Pwr_DACJitter);

% If Per_Cell is ON, Calc SNR
if Per_Cell

    BW_Pwr_CellJitter = sum(OS_psdestx_Bit(Bin_L:Bin_U));
    Signal_Pwr_CellJitter = sum(OS_psdestx_Bit(Bin:Bin+2));
    Noise_Pwr_CellJitter = BW_Pwr_CellJitter - ...
        Signal_Pwr_CellJitter;
    SNR_CellJitter = 10*log10(Signal_Pwr_CellJitter/...
        Noise_Pwr_CellJitter);
else

    SNR_CellJitter = 0;
end

% Clear variables for memory
if Per_Cell
    clear OS_Fxx OS_Fxx2 OS_Fxx1 OS_Fxx_add Fxx
    clear OS_psdestx_add OS_psdestx_Bit OS_psdestx2 OS_psdestx1
end

% Output Simulation Data

disp(' ');
disp('--- Run Data ------');
disp(['Eff. Sample Rate: ' num2str(Fs/1e9) ' GHz']);
disp(['Single DAC Rate: ' num2str(Fs/2/1e9) ' GHz']);
disp(['Num. of Samples: ' num2str(Num_Samples)]);
disp(['Jitter OSR: ' num2str(OSR)]);
disp(['DS Resolution: ' num2str(Resolution)]);
disp(['DAC Resolution: ' num2str(Bits)]);
disp(['Output Bin: ' num2str(Bin)]);
disp(['Signal Amplitude: ' num2str(Signal_Amplitude)]);

if Per_Cell == 1
    disp('Per Cell Jitter: On');
elseif Per_Cell == 0
    disp('Per Cell Jitter: Off');
else

end
disp('Per Cell Jitter: ?');
end

disp('');
disp('------ SNR ------');

if Bin < Bin_L+2 || Bin > Bin_U-2
disp('Signal is not within the SNR BW')
else
    disp(['SNR Bandwidth: ' num2str(Fs/4 * SNR_BW/1e9) ' GHz']);
    disp(['Jitter Sigma: ' num2str(Sigma*1e12) ' ps']);
    disp(['Ideal SNR: ' num2str(SNR_Ideal) ' dB']);
    disp(['Sys Jitter SNR: ' num2str(SNR_AllJitter) ' dB']);
    disp(['DAC Jitter SNR: ' num2str(SNR_DACJitter) ' dB']);
    if Per_Cell
        disp(['Cell Jitter SNR: ' num2str(SNR_CellJitter) ' dB']);
    end
end

disp('');
disp('');
end
Appendix C: Testing Methodology - Rev. 2

The following appendix gives an overview of the testing of the final ΔΣ DAC revision, highlighting the setup and a few of the tests performed on the proposed design.

C.1 Test Equipment

The DAC testing, again pictured in Fig. C.1, is accomplished with the following equipment:

- A VC7215 Virtex 7 Evaluation Board providing high-speed data input via fourteen LVDS channels
- A signal generator providing the FPGA clock reference (-10 dBm)
- A second signal generator for the external (x2) DAC clock. Due to high frequency attenuation, the clock input power must scale with frequency.
- Local computer for FPGA control and programming
- SPI register control enabled by a Spartan 6 FPGA on an Opal Kelly board.
- High frequency, wide-band balun for (optional) differential measurement
- PXA Signal Analyzer for spectrum measurement
- DC Power supplies
Figure C.1: Test setup of the interleaved ΔΣ DAC.

C.2 Performance and Verification Tests

DC Testing

Board and chip integrity can be ensured with a DC test. By connecting the power supplies to the voltage header, shown in Fig. C.2, the board connectivity can be validated by comparing the current draw to the expected values given in Table C.1. Due to clock buffering, the digital current draw scales with clock frequency and requires an increased digital supply to overcome on-board voltage droop due to the ohmic loss of the power cables.

Data and Clock Testing

On-chip clock functionality can be verified by bringing a divided down clock signal back off-chip. The divide-by ratio is control by the CLK_CTL1 register through the SPI interface and can be observed on the differential SMA outputs on the south of the
board. Once the on-chip clock is functional, the chip can receive LVDS data inputs (which should be 100 - 200 mV in amplitude). The data inputs can be routed off-chip one-at-a-time using the LOOPBACKx registers to control channel selection with power jumpers in-place to enable the on-chip LVDS transmitters. The data for one channel can be viewed on the vertical SMAs. This is critical for confirming data latching and proper alignment before the integrated DSM (Fig. C.3).
Calibration

Bypassing the on-chip DSM, data can be sent directly to the DAC by setting the DS_CTL register. This is useful for calibration since each unary DAC cell can be independently toggled and its output power compared to a reference. The current of each cell can then be adjusted such that each DAC cell has the same magnitude. This is accomplished by setting CAL_DAC_DATA_SELx for manual input and then adjusting the currents via CAL_DAC_DATAx_S. Likewise the timing of each cell can be adjusted by setting CAL_TADJ_DATA_SELx to manual and then tuning through the CAL_TADJ_DATAx_S registers. To test for intrinsic accuracy, the calibration DACs can be turned off, but calibration is aided with a mid-level starting value.

General timing between the parallel DACs can be altered with the individual timing adjustment listed above. In addition, the timing between channels can also be control with the CLK_CTLx and CLK_ALIGN registers, demonstrating the dependence
of SIRR on inter-channel timing. Since this inter-channel timing is controlled with switched varactors, small adjustments to the clock voltage supply also tune the timing between channels, allowing the two DACs to be more precisely aligned. While enabling fine tuning, this feature adds jitter to the DAC design and is to be removed in subsequent revisions. The diagram for this testing is given in Fig. C.4.

![Figure C.4: Functional diagram of the DSM by-pass testing.](image)

**DSM By-pass Testing**

After calibration, pre-computed ∆Σ modulated data can be passed directly to the DAC using the same configuration as that of calibration. This testing allows for SIRR, SFDR, IM3, and ACLR performance extraction without directly passing through the on-chip DSM, greatly increasing testing efficiency. Performance before and after calibration can be shown in addition to single versus interleaved DAC operation.
DSM Testing

Once the DAC operation has been verified, the DSM can be tested. This requires two 7 channel LVDS inputs consisting of doubled-data-rate sine wave samples. The sample bits must be pre-aligned off-chip to account for the pipelined architecture. This can be confirmed on-chip using the previously mentioned loopback testing to directly observed the data. To adjust data alignment, the \texttt{DATA\_ARR} register can be used to properly position the MSB and LSB of each channel. During DSM operation, static offsets can be added to the incoming samples using the \texttt{DS\_OFFSETx} registers. This functionality is to ensure the DSM bit cells do not experience overflow. The testing is depicted by the digram given in Fig. C.5.

![Functional diagram of the full DSM and DAC testing.](image)

Figure C.5: Functional diagram of the full DSM and DAC testing.
## Appendix D: SPI Register Definitions

### D.1 Register Map

Table D.1: Register Map for DAC Serial Interface

<table>
<thead>
<tr>
<th>Address</th>
<th>Name</th>
<th>R/W</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x0000 - 0x0001</td>
<td>DAC_CTLx(^1)</td>
<td>R/W</td>
<td>DAC calibration control bits</td>
</tr>
<tr>
<td>0x0004</td>
<td>DAC_STATUS</td>
<td>R</td>
<td>DAC calibration status bits</td>
</tr>
<tr>
<td>0x0008 - 0x0009</td>
<td>LOOPBACKx(^1)</td>
<td>R/W</td>
<td>Data loopback test: channel select</td>
</tr>
<tr>
<td>0x0010 - 0x001D</td>
<td>DATA_RX_ADJx(^1)</td>
<td>R/W</td>
<td>Data RX phase adjustment</td>
</tr>
<tr>
<td>0x0040</td>
<td>CAL_CUT_L</td>
<td>R</td>
<td>Generated cell-under-test for cal.</td>
</tr>
<tr>
<td>0x0050</td>
<td>CAL_LO_ADJ</td>
<td>R/W</td>
<td>Phase adjust for calibration LO</td>
</tr>
<tr>
<td>0x0051</td>
<td>CAL_LO_DATA_L</td>
<td>R</td>
<td>Generated data pattern for calibration LO</td>
</tr>
<tr>
<td>0x0052</td>
<td>CAL_LO_DATA_S</td>
<td>R/W</td>
<td>Manual data pattern for calibration LO</td>
</tr>
<tr>
<td>0x0058</td>
<td>CAL_MEAS_DELAY</td>
<td>R/W</td>
<td>Delay before taking measurement</td>
</tr>
<tr>
<td>0x0060</td>
<td>CAL_OS_L</td>
<td>R</td>
<td>Generated measurement offset</td>
</tr>
<tr>
<td>0x0061</td>
<td>CAL_GE_L</td>
<td>R</td>
<td>Generated gain error measurement</td>
</tr>
<tr>
<td>0x0064</td>
<td>CAL_TARGET_A</td>
<td>R/W</td>
<td>Amplitude calibration target</td>
</tr>
<tr>
<td>0x0065</td>
<td>CAL_TARGET_T</td>
<td>R/W</td>
<td>Timing calibration target</td>
</tr>
<tr>
<td>0x0070 - 0x007D</td>
<td>CAL_I_RESx(^1)</td>
<td>R</td>
<td>Result of current measurement</td>
</tr>
<tr>
<td>0x00C0</td>
<td>CAL_DATA_SELx(^1)</td>
<td>R/W</td>
<td>Calibration data pattern select</td>
</tr>
<tr>
<td>0x00C8</td>
<td>CAL_DAC_DATA_SELx(^1)</td>
<td>R/W</td>
<td>Cal. DAC data (Manual Over-ride)</td>
</tr>
<tr>
<td>0x00CC</td>
<td>CAL_TADJ_DATA_SELx(^1)</td>
<td>R/W</td>
<td>Cal. timing data (Manual Over-ride)</td>
</tr>
<tr>
<td>0x00DD - 0x00DF</td>
<td>CAL_DATAx_L(^1)</td>
<td>R</td>
<td>Generated data patterns for cal.</td>
</tr>
<tr>
<td>0x0100 - 0x010F</td>
<td>CAL_DAC_DATAx_L(^1)</td>
<td>R</td>
<td>Generated data for each CALDAC</td>
</tr>
<tr>
<td>0x0130 - 0x013F</td>
<td>CAL_TADJ_DATAx_L(^1)</td>
<td>R</td>
<td>Generated timing adjustments</td>
</tr>
<tr>
<td>0x0170 - 0x017D</td>
<td>CAL_T_RESx(^1)</td>
<td>R</td>
<td>Result of timing measurement</td>
</tr>
<tr>
<td>0x0240</td>
<td>CAL_CUT_S</td>
<td>R/W</td>
<td>Manual cell-under-test for cal.</td>
</tr>
<tr>
<td>0x0260</td>
<td>CAL_OS_S</td>
<td>R/W</td>
<td>Manual measurement offset</td>
</tr>
<tr>
<td>0x0261</td>
<td>CAL_GE_S</td>
<td>R/W</td>
<td>Manual gain error adjust</td>
</tr>
<tr>
<td>0x02D0 - 0x02DF</td>
<td>CAL_DATAx_S(^1)</td>
<td>R/W</td>
<td>Manual data patterns for cal.</td>
</tr>
<tr>
<td>0x0300 - 0x030F</td>
<td>CAL_DAC_DATAx_S(^1)</td>
<td>R/W</td>
<td>Manual data for each CALDAC</td>
</tr>
<tr>
<td>0x0330 - 0x033F</td>
<td>CAL_TADJ_DATAx_S(^1)</td>
<td>R/W</td>
<td>Manual data for timing adj.</td>
</tr>
<tr>
<td>0x0400</td>
<td>DS_CTL</td>
<td>R/W</td>
<td>DS modulator control bits</td>
</tr>
<tr>
<td>0x0401 - 0x0402</td>
<td>CLK_CTLx(^1)</td>
<td>R/W</td>
<td>Clock phase control bits</td>
</tr>
<tr>
<td>0x0410 - 0x0415</td>
<td>DS_OFFSETx(^1)</td>
<td>R/W</td>
<td>DS modulator DC offset</td>
</tr>
<tr>
<td>0x0440</td>
<td>CLK_ALIGN</td>
<td>R/W</td>
<td>CML and CMOS clock alignment</td>
</tr>
<tr>
<td>0x0450</td>
<td>DATA_ARR</td>
<td>R/W</td>
<td>Interleaved data arrangement</td>
</tr>
</tbody>
</table>

\(^1\) For registers listed with multiple addresses, the register name is specified by \(x = 0 \ldots N - 1\), where \(N\) is the number of 16-bit addresses.
D.2 Register Definitions

DAC_CTLx Register

The DAC_CTLx registers contain various bits to set the operation mode of the DAC.

<table>
<thead>
<tr>
<th>Bit Field</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15 - 12 NS_SEL</td>
<td>0x0</td>
<td>Number of calibration samples per measurement (N_s = 2^{16 + n_{SEL}})</td>
</tr>
<tr>
<td>11 - 10 Reserved</td>
<td>0x0</td>
<td>Reserved</td>
</tr>
<tr>
<td>9 MANCUT</td>
<td>0x0</td>
<td>Enable manual overried for CUT</td>
</tr>
<tr>
<td>8 LOSEL</td>
<td>0x0</td>
<td>Calibration LO data select. Selects between manual or generated LO data patterns</td>
</tr>
<tr>
<td>7 CSSEL</td>
<td>0x0</td>
<td>Cell select override. Allows for manually overriding the cell select in order to load in calibration data.</td>
</tr>
<tr>
<td>6 CDEN</td>
<td>0x0</td>
<td>Calibration data enable. Bypasses input data and drives the DAC with calibration data.</td>
</tr>
<tr>
<td>5 - 4 CLKDIV</td>
<td>0x0</td>
<td>Clock select for calibration circuitry* (\left( f_{CAL} = \frac{f_{DATA}}{2^{5 + CLKDIV}} \right) )</td>
</tr>
<tr>
<td>3 CALMIX</td>
<td>0x0</td>
<td>Enable the calibration mixer. (0b =) Mixer enabled (1b =) Mixer disabled</td>
</tr>
<tr>
<td>2 CALST</td>
<td>0x0</td>
<td>Start the full calibration process when set from 0b to 1b. Automatically set to 0b when calibration is finished.</td>
</tr>
<tr>
<td>1 MEAS</td>
<td>0x0</td>
<td>Start a single calibration measurement when set from 0b to 1b.</td>
</tr>
<tr>
<td>0 CALEN</td>
<td>0x0</td>
<td>Enable calibration circuitry (1b =) Calibration circuitry enabled (0b =) Calibration circuitry disabled</td>
</tr>
</tbody>
</table>

* Divided clock is coupling to the output; changing divide reduces coupling
Table D.3: DAC_CTL1 Register Description

<table>
<thead>
<tr>
<th>Bit Field</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15 - 0</td>
<td>Reserved</td>
<td>0x0 Reserved</td>
</tr>
</tbody>
</table>

DAC_STATUS Register

The DAC_STATUS register is a read-only register that contains status bits pertaining to the current state of the calibration.

Figure D.2: DAC_CTL1 Register Fields

Figure D.3: DAC_STATUS Register Fields
Table D.4: DAC_STATUS Register Description

<table>
<thead>
<tr>
<th>Bit Field</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15 - 1</td>
<td>0x0</td>
<td>Reserved</td>
</tr>
<tr>
<td>0</td>
<td>BUSY</td>
<td>0x0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Status of DAC calibration circuitry. Do not write to any registers if a calibration or measurement is in progress.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0b = Idle</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1b = Calibration in progress</td>
</tr>
</tbody>
</table>

LOOPBACKx Register

The LOOPBACKx registers select the data channels for loopback.

<table>
<thead>
<tr>
<th>Bit</th>
<th>Field</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td></td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td></td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td></td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td></td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td></td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>LBSEL</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure D.4: LOOPBACKx Register Fields

Table D.5: LOOPBACKx Register Description

<table>
<thead>
<tr>
<th>Bit Field</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15 - 4</td>
<td>0x0</td>
<td>Reserved</td>
</tr>
<tr>
<td>3-0</td>
<td>LBSEL</td>
<td>0x0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Select the data channel for loopback</td>
</tr>
</tbody>
</table>

DATA_RX_ADJx Register

The DATA_RX_ADJx registers are a group of registers that control the phases of the incoming data to ensure that each data channel is properly aligned with the data clock.
Figure D.5: DATA_RX_ADJx Register Fields

Table D.6: DATA_RX_ADJx Register Description

<table>
<thead>
<tr>
<th>Bit Field</th>
<th>Field</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15 - 4</td>
<td>Reserved</td>
<td>0x0</td>
<td>Reserved</td>
</tr>
<tr>
<td>3 - 0</td>
<td>ADJ</td>
<td>0x8</td>
<td>RX phase adjustment for data bit x</td>
</tr>
</tbody>
</table>

CAL_CUT_L Register

The CAL_CUT_L register holds the cell-under-test as generated by the calibration logic.

Figure D.6: CAL_CUT_L Register Fields

Table D.7: CAL_CUT_L Register Description

<table>
<thead>
<tr>
<th>Bit Field</th>
<th>Field</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15 - 4</td>
<td>Reserved</td>
<td>0x0</td>
<td>Reserved</td>
</tr>
<tr>
<td>3 - 0</td>
<td>CUTIDX</td>
<td>0x0</td>
<td>Index for the CUT</td>
</tr>
</tbody>
</table>
**CAL\_CUT\_S Register**

The `CAL\_CUT\_S` register allows for manually specify a cell-under-test for performing a calibration measurement. Setting the corresponding cell index in this register allows for manually performing a single calibration on any of the DAC cells without having to run a full calibration.

![CAL\_CUT\_S Register Fields](image)

**Table D.8: CAL\_CUT\_S Register Description**

<table>
<thead>
<tr>
<th>Bit Field</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15 - 4</td>
<td>Reserved</td>
<td>0x0</td>
</tr>
<tr>
<td>3 - 0</td>
<td>CUTIDX</td>
<td>0x0</td>
</tr>
</tbody>
</table>

**CAL\_LO\_ADJ Register**

The `CAL\_LO\_ADJ` register adjusts the phase of the LO for the calibration mixer and is used to align the LO with the phase of the cell-under-test.

![CAL\_LO\_ADJ Register Fields](image)

**Figure D.8: CAL\_LO\_ADJ Register Fields**
Table D.9: CAL_LO_ADJ Register Description

<table>
<thead>
<tr>
<th>Bit Field</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15 - 4</td>
<td>Reserved</td>
<td>0x0 Reserved</td>
</tr>
<tr>
<td>3 - 0</td>
<td>ADJ</td>
<td>0x8 Phase adjustment for the calibration mixer LO</td>
</tr>
</tbody>
</table>

CAL_LO_DATA_L Register

The CAL_LO_DATA_L register holds a data pattern that is used to generate the LO for the calibration mixer.

```
  15  14  13  12  11  10   9   8
    DATAL
     7  6  5  4  3  2  1  0
    DATAL
```

Figure D.9: CAL_LO_DATA_L Register Fields

Table D.10: CAL_LO_DATA_L Register Description

<table>
<thead>
<tr>
<th>Bit Field</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15 - 0</td>
<td>DATAL</td>
<td>0x0 Generated data pattern for cell x</td>
</tr>
</tbody>
</table>

CAL_LO_DATA_S Register

The CAL_LO_DATA_S register is used to manually set a data pattern used to generate the calibration mixer LO.

```
  15  14  13  12  11  10   9   8
    DATAS
     7  6  5  4  3  2  1  0
    DATAS
```

Figure D.10: CAL_LO_DATA_S Register Fields

139
Table D.11: CAL_LO_DATA_S Register Description

<table>
<thead>
<tr>
<th>Bit</th>
<th>Field</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15 - 0</td>
<td>DATAS</td>
<td>0x0</td>
<td>Manual data pattern for cell x</td>
</tr>
</tbody>
</table>

CAL_MEAS_DELAY Register

The CAL_MEAS_DELAY register sets the number of calibration cycles to wait before a measurement is taken. This allows time for the ΔΣ modulator to settle before measuring the output of the DAC.

\[
\begin{array}{cccccccc}
15 & 14 & 13 & 12 & 11 & 10 & 9 & 8 \\
\hline \\
\text{CALDLY} \\
\end{array}
\]

\[
\begin{array}{cccccccc}
7 & 6 & 5 & 4 & 3 & 2 & 1 & 0 \\
\hline \\
\text{CALDLY} \\
\end{array}
\]

Figure D.11: CAL_MEAS_DELAY Register Fields

Table D.12: CAL_MEAS_DELAY Register Description

<table>
<thead>
<tr>
<th>Bit</th>
<th>Field</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15 - 0</td>
<td>CALDLY</td>
<td>0x0FF</td>
<td>Number of delay cycles</td>
</tr>
</tbody>
</table>

CAL_OS_L Register

The CAL_OS_L register holds the measured offset of the calibration measurement circuitry that is then subtracted from the measurement results.
CALOSL Register

Table D.13: CAL_OS_L Register Description

<table>
<thead>
<tr>
<th>Bit</th>
<th>Field</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15-14</td>
<td>Reserved</td>
<td>0x0</td>
<td>Reserved</td>
</tr>
<tr>
<td>13-0</td>
<td>CALOSL</td>
<td>0x0</td>
<td>Measured offset</td>
</tr>
</tbody>
</table>

CALOS_S Register

The CALOS_S register holds a manual offset that can be optionally subtracted from the calibration measurement.

Table D.14: CAL_OS_S Register Description

<table>
<thead>
<tr>
<th>Bit</th>
<th>Field</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15-14</td>
<td>Reserved</td>
<td>0x0</td>
<td>Reserved</td>
</tr>
<tr>
<td>13-0</td>
<td>CALOSS</td>
<td>0x0</td>
<td>Manual offset</td>
</tr>
</tbody>
</table>
**CAL\_GE\_L Register**

The CAL\_GE\_L register holds the measured gain error of the calibration measurement circuitry that is then de-embedded from the calibration measurement.

![Figure D.14: CAL\_GE\_L Register Fields](image)

<table>
<thead>
<tr>
<th>Bit Field</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15 - 14</td>
<td>Reserved</td>
<td>0x0</td>
</tr>
<tr>
<td>13 - 0</td>
<td>CALGEL</td>
<td>0x0</td>
</tr>
</tbody>
</table>

**CAL\_GE\_S Register**

The CAL\_GE\_S register holds a manual gain error compensation value that can be optionally de-embedded from the calibration measurement.

![Figure D.15: CAL\_GE\_S Register Fields](image)

<table>
<thead>
<tr>
<th>Bit Field</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15-14</td>
<td>Reserved</td>
<td>0x0</td>
</tr>
<tr>
<td>13 - 0</td>
<td>CALGES</td>
<td>0x0</td>
</tr>
</tbody>
</table>
CAL_TARGET_A Register

The CAL_TARGET_A register sets the target amplitude to which each DAC current cell is calibrated.

```
 15  14  13  12  11  10  9  8
  N/A ATGT
  7  6  5  4  3  2  1  0
```

Figure D.16: CAL_TARGET_A Register Fields

<table>
<thead>
<tr>
<th>Bit Field</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15 - 14</td>
<td>Reserved</td>
<td>0x0</td>
</tr>
<tr>
<td>13 - 0</td>
<td>ATGT</td>
<td>0x2000</td>
</tr>
</tbody>
</table>

Table D.17: CAL_TARGET_A Register Description

CAL_TARGET_T Register

The CAL_TARGET_T register sets the target timing offset to which all cells are calibrated.

```
  15  14  13  12  11  10  9  8
  N/A TTGT
  7  6  5  4  3  2  1  0
```

Figure D.17: CAL_TARGET_T Register Fields

<table>
<thead>
<tr>
<th>Bit Field</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15 - 14</td>
<td>Reserved</td>
<td>0x0</td>
</tr>
<tr>
<td>13 - 0</td>
<td>TTGT</td>
<td>0x0</td>
</tr>
</tbody>
</table>

Table D.18: CAL_TARGET_T Register Description
CAL_I_RESx Register

The CAL_I_RESx registers are a group of read-only registers that contain the results of the calibration current measurement for each cell.

```
+---------+---------+---------+---------+---------+---------+---------+---------+  
| 15 - 14 | 13 - 0  |         |         |         |         |         |         |  
| Reserved | IRES    | Reserved | Reserved |         |         |         |         |  
| 0x0      | 0x0     | Current | measurement for cell x. |  
|         |  
```

Figure D.18: CAL_I_RESx Register Fields

Table D.19: CAL_I_RESx Register Description

<table>
<thead>
<tr>
<th>Bit</th>
<th>Field</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15 - 14</td>
<td>Reserved</td>
<td>0x0</td>
<td>Reserved</td>
</tr>
<tr>
<td>13 - 0</td>
<td>IRES</td>
<td>0x0</td>
<td>Current measurement result for cell x.</td>
</tr>
</tbody>
</table>

CAL_T_RESx Register

The CAL_T_RESx registers are a group of read-only registers that contain the results of the calibration timing measurement for each cell.

```
+---------+---------+---------+---------+---------+---------+---------+---------+  
| 15 - 14 | 13 - 0  |         |         |         |         |         |         |  
| Reserved | TRES    | Reserved | Reserved |         |         |         |         |  
| 0x0      | 0x0     | Timing | measurement result for cell x. |  
|         |  
```

Figure D.19: CAL_T_RESx Register Fields

Table D.20: CAL_T_RESx Register Description

<table>
<thead>
<tr>
<th>Bit</th>
<th>Field</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15 - 14</td>
<td>Reserved</td>
<td>0x0</td>
<td>Reserved</td>
</tr>
<tr>
<td>13 - 0</td>
<td>TRES</td>
<td>0x0</td>
<td>Timing measurement result for cell x.</td>
</tr>
</tbody>
</table>

144
CAL_DATA_SELx Register

The CAL_DATA_SELx registers are used to select between the CALDAC data that are generated by the calibration logic and the data that are manually specified through the SPI.

<table>
<thead>
<tr>
<th>15</th>
<th>14</th>
<th>13</th>
<th>12</th>
<th>11</th>
<th>10</th>
<th>9</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>DS15</td>
<td>DS14</td>
<td>DS13</td>
<td>DS12</td>
<td>DS11</td>
<td>DS10</td>
<td>DS9</td>
<td>DS8</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>DS7</td>
<td>DS6</td>
<td>DS5</td>
<td>DS4</td>
<td>DS3</td>
<td>DS2</td>
<td>DS1</td>
<td>DS0</td>
</tr>
</tbody>
</table>

Figure D.20: CAL_DATA_SEL0 Register Fields

Table D.21: CAL_DATA_SEL0 Register Description

<table>
<thead>
<tr>
<th>Bit Field</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15 - 0 DSn</td>
<td>0x0</td>
<td>Data select for cell n</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0b = Use pattern generated by calibration logic</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1b = Use pattern specified by SPI</td>
</tr>
</tbody>
</table>

CAL_DAC_DATA_SELx Register

The CAL_DAC_DATA_SELx registers are used to select between the CALDAC data that are generated by the calibration logic and data that are manually specified through the SPI.
Table D.22: CAL_DAC_DATA_SEL0 Register Description

<table>
<thead>
<tr>
<th>Bit</th>
<th>Field</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>DSn</td>
<td>0x0</td>
<td>Data select for cell $n$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$0b = \text{Use pattern generated by calibration logic}$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$1b = \text{Use pattern specified by SPI}$</td>
</tr>
</tbody>
</table>

CAL_TADJ_DATA_SELx Register

The CAL_TADJ_DATA_SELx registers are used to select between timing adjustments that are generated by the calibration logic and patterns that are manually specified through the SPI.
Table D.23: CAL_TADJ_DATA_SEL0 Register Description

<table>
<thead>
<tr>
<th>Bit Field</th>
<th>Reset Description</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15 - 0</td>
<td>0x0</td>
<td>DSn</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0b = Use pattern generated by calibration logic</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1b = Use pattern specified by SPI</td>
</tr>
</tbody>
</table>

CAL_DATAx_L Register

The CAL_DATAx_L registers are a group of registers that hold a data pattern for each cell that is generated by the calibration logic. During a calibration measurement, each DAC cell will cycle through its corresponding data pattern.

Figure D.23: CAL_DATAx_L Register Fields

Table D.24: CAL_DATAx_L Register Description

<table>
<thead>
<tr>
<th>Bit Field</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15 - 0</td>
<td>0x0</td>
<td>DATAL</td>
</tr>
<tr>
<td>7 - 0</td>
<td></td>
<td>DATAL</td>
</tr>
</tbody>
</table>

CAL_DATAx_S Register

The CAL_DATAx_S registers are a group of registers that are used to manually set a data pattern for each cell. During a calibration measurement, each DAC cell will cycle through its corresponding data pattern.
CAL\_DAC\_DATAx\_L Register

The CAL\_DAC\_DATAx\_L registers are a group of registers contain the automatically generated calibration DAC data that sets the calibration current for each cell via the calibration DACs.

![Figure D.25: CAL\_DAC\_DATAx\_L Register Fields](image)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Field</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>0</td>
<td>0x0</td>
<td>Manual data pattern for cell x</td>
</tr>
</tbody>
</table>

Table D.26: CAL\_DAC\_DATAx\_L Register Description

<table>
<thead>
<tr>
<th>Bit</th>
<th>Field</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>7</td>
<td>0x0</td>
<td>Reserved</td>
</tr>
<tr>
<td>8</td>
<td>DATA</td>
<td>0x8</td>
<td>Data for the calibration DAC for cell x</td>
</tr>
</tbody>
</table>

148
CAL_DAC_DATAx_S Register

The CAL_DAC_DATAx_S registers are a group of registers that contain the manually specified calibration DAC data that sets the calibration current for each cell via the calibration DACs.

<table>
<thead>
<tr>
<th>Bit</th>
<th>Field</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15 - 7</td>
<td>Reserved</td>
<td>0x0</td>
<td>Reserved</td>
</tr>
<tr>
<td>8 - 0</td>
<td>DATA</td>
<td>0x8</td>
<td>Data for the calibration DAC for cell x</td>
</tr>
</tbody>
</table>

CAL_TADJ_DATAx_L Register

The CAL_TADJ_DATAx_L registers are a group of registers that contain the automatically generated bits that set the timing adjustment for each cell via the local RZ drivers.
Table D.28: CAL_TADJ_DATAx_L Register Description

<table>
<thead>
<tr>
<th>Bit Field</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15 - 5</td>
<td>0x0</td>
<td>Reserved</td>
</tr>
<tr>
<td>4 - 0</td>
<td>0x8</td>
<td>Data for the calibration DAC for cell x</td>
</tr>
</tbody>
</table>

CAL_TADJ_DATAx_S Register

The CAL_TADJ_DATAx_S registers are a group of registers that contain the manually specified bits that set the timing adjustment for each cell via the local RZ drivers.

<table>
<thead>
<tr>
<th>Bit</th>
<th>Field</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>14</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>13</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>12</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>11</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>10</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>9</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>8</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>7</td>
<td>N/A</td>
<td>N/A</td>
<td>DATA</td>
</tr>
<tr>
<td>6</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>5</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>4</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>3</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>2</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>1</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>0</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>

Figure D.28: CAL_TADJ_DATAx_S Register Fields

Table D.29: CAL_TADJ_DATAx_S Register Description

<table>
<thead>
<tr>
<th>Bit Field</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15 - 5</td>
<td>0x0</td>
<td>Reserved</td>
</tr>
<tr>
<td>4 - 0</td>
<td>0x8</td>
<td>Data for the calibration DAC for cell x</td>
</tr>
</tbody>
</table>

DS_CTL Register

The DS_CTL register holds the control bits for the on chip DS Modulators.

<table>
<thead>
<tr>
<th>Bit</th>
<th>Field</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>14</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>13</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>12</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>11</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>10</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>9</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>8</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>7</td>
<td>N/A</td>
<td>N/A</td>
<td>DAC_SEL</td>
</tr>
<tr>
<td>6</td>
<td>N/A</td>
<td>N/A</td>
<td>BP/HP</td>
</tr>
<tr>
<td>5</td>
<td>N/A</td>
<td>N/A</td>
<td>ORDER</td>
</tr>
<tr>
<td>4</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>3</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>2</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>1</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>0</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>

Figure D.29: DS_CTL Register Fields

150
Table D.30: DS_CTL Register Description

<table>
<thead>
<tr>
<th>Bit</th>
<th>Field</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15 - 5</td>
<td>Reserved</td>
<td>0x0</td>
<td>Reserved</td>
</tr>
<tr>
<td>3</td>
<td>DAC_SEL</td>
<td>0x0</td>
<td>Select between DAC0 and DAC1 for calibration</td>
</tr>
<tr>
<td>2</td>
<td>BP/HP</td>
<td>0x0</td>
<td>Highpass or Bandpass DS Modulation</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0b = Highpass</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1b = Bandpass</td>
</tr>
<tr>
<td>1-0</td>
<td>ORDER</td>
<td>0x0</td>
<td>DS Modulation Order</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>00b = Pass Through</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>01b = 1st Order</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>10b = 2nd Order</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>11b = 3rd Order</td>
</tr>
</tbody>
</table>

CLK_CTLx Register

The CLK_CTLx registers are a group of registers that control the on-chip clock phase adjustment.

Figure D.30: CLK_CTLx Register Fields

Table D.31: CLK_CTLx Register Description

<table>
<thead>
<tr>
<th>Bit</th>
<th>Field</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>Reserved</td>
<td>0x0</td>
<td>Reserved</td>
</tr>
<tr>
<td>13</td>
<td>CLKOUTDIV</td>
<td>0x0</td>
<td>Clock Output Divide</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Only one on at a time</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Only in CLK_CTL1</td>
</tr>
<tr>
<td>9</td>
<td>CMLx_PHADJ</td>
<td>0x0</td>
<td>CML Clock Phase Adjustment</td>
</tr>
<tr>
<td>7</td>
<td>INVx_PHADJ</td>
<td>0x0</td>
<td>Inverter (Digital) Fine Phase Adjustment</td>
</tr>
</tbody>
</table>
**DS_OFFSETx Register**

The DS_OFFSETx registers are a group of registers that hold the DC offset for a stage of the DS Modulator.

<table>
<thead>
<tr>
<th>Bit</th>
<th>Field</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15 - 14</td>
<td>Reserved</td>
<td>0x0</td>
<td>Reserved</td>
</tr>
<tr>
<td>13</td>
<td>OFFSET</td>
<td>0x2000</td>
<td>DC offset value</td>
</tr>
</tbody>
</table>

- Figure D.31: DS_OFFSETx Register Fields

**CLK_ALIGN Register**

The CLK_ALIGN controls the alignment between CML and CMOS clocks.

<table>
<thead>
<tr>
<th>Bit</th>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>CML1 ADJ</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>CML0 ADJ</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>CML ST2</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>INV</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>CML ST1</td>
<td></td>
</tr>
</tbody>
</table>

- Figure D.32: CLK_ALIGN Register Fields
Table D.33: CLK_ALIGN Register Description

<table>
<thead>
<tr>
<th>Bit Field</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15 - 7</td>
<td>0x0</td>
<td>Reserved</td>
</tr>
<tr>
<td>6 - 5</td>
<td>0x0</td>
<td>DAC1 CML clock adjust</td>
</tr>
<tr>
<td>4 - 3</td>
<td>0x0</td>
<td>DAC0 CML clock adjust</td>
</tr>
<tr>
<td>2</td>
<td>0x0</td>
<td>CML Stage two 180° phase shift</td>
</tr>
<tr>
<td>1</td>
<td>0x0</td>
<td>Digital (inverter buffer) 180° phase shift</td>
</tr>
<tr>
<td>0</td>
<td>0x0</td>
<td>CML Stage one 180° phase shift</td>
</tr>
</tbody>
</table>

DATA_ARR Register

The DATA_ARR toggles the arrangement (LSB/MSB) of the incoming serial data.

<table>
<thead>
<tr>
<th>Bit</th>
<th>Field</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>Reserved</td>
<td>0x0</td>
<td>Reserved</td>
</tr>
<tr>
<td>14</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>DATAARR</td>
<td>0x0</td>
<td>DATA arrangement control</td>
</tr>
<tr>
<td>12</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure D.33: DATA_ARR Register Fields

Table D.34: DATA_ARR Register Description

<table>
<thead>
<tr>
<th>Bit</th>
<th>Field</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15 - 14</td>
<td>Reserved</td>
<td>0x0</td>
<td>Reserved</td>
</tr>
<tr>
<td>13 - 0</td>
<td>DATAARR</td>
<td>0x0</td>
<td>DATA arrangement control</td>
</tr>
</tbody>
</table>

153
Bibliography


154


