

**ERROR MODELING, SELF-CALIBRATION AND DESIGN
OF PIPELINED ANALOG TO DIGITAL CONVERTERS**

Volume I

A Dissertation

by

ERIC GEORGES SOENEN

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of
DOCTOR OF PHILOSOPHY

May 1992

Major Subject: Electrical Engineering

© 1992

ERIC GEORGES SOENEN
ALL RIGHTS RESERVED

**ERROR MODELING, SELF-CALIBRATION AND DESIGN
OF PIPELINED ANALOG TO DIGITAL CONVERTERS**

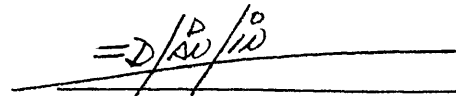
Volume I

A Dissertation
by
ERIC GEORGES SOENEN

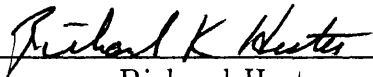
Approved as to style and content by:



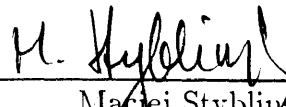
Randall Geiger
(Co-Chair of Committee)



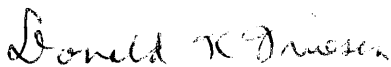
Edgar Sanchez-Sinencio
(Co-Chair of Committee)



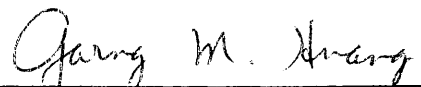
Richard Hester
(Member)



Maciej Styblinski
(Member)



Donald Friesen
(Member)



Jo Howze
(Head of Department)

May 1992

ABSTRACT

Error Modeling, Self-Calibration and Design
of Pipelined Analog to Digital Converters. (May 1992)

Eric Georges Soenen, B.S., Katholieke Universiteit Leuven, Belgium

M.S., Texas A&M University

Chair of Advisory Committee: Dr. Randall L. Geiger
Dr. Edgar Sanchez-Sinencio

As the field of signal processing accelerates toward the use of high performance digital techniques, there is a growing need for increasingly fast and accurate analog to digital converters.

Three highly visible examples of this trend originated in the last decade. The advent of the compact disc revolutionized the way high-fidelity audio is stored, reproduced, recorded and processed. Digital communication links, fiber optic cables and in the near future ISDN networks (Integrated Services Digital Network) are steadily replacing major portions of telephone systems. Finally, video-conferencing, multi-media computing and currently emerging high definition television (HDTV) systems rely more and more on real-time digital data compression and image enhancing techniques.

All these applications rely on analog to digital conversion. In the field of digital audio, the required conversion accuracy is high, but the conversion speed limited (16 bits, 2 x 20 kHz signal bandwidth). In the field of image processing, the required accuracy is less, but the data conversion speed high (8-10 bits, 5-20MHz bandwidth). New applications keep pushing for increasing conversion rates and simultaneously higher accuracies. This dissertation discusses new analog to digital converter architectures that could accomplish this.

As a consequence of the trend towards digital processing, prominent analog designers throughout the world have engaged in very active research on the topic of data conversion. Unfortunately, literature has not always kept up. At the time

of this writing, it seemed rather difficult to find detailed fundamental publications about analog to digital converter design. This dissertation represents a modest attempt to remedy this situation. It is hoped that anyone with a back-ground in analog design could go through this work and pick up the fundamentals of converter operation, as well as a number of more advanced design techniques.

To the memory of Peter M. VanPeteghem

ACKNOWLEDGEMENTS

The work that went into this dissertation is dedicated to the memory of Dr. Peter VanPeteghem. Dr. VanPeteghem was the advisor for my Master's degree and was also to be the advisor for my subsequent Ph.D. degree. He was a great teacher, respected by all for his tremendous knowledge, his enthusiasm and his deep human qualities. His presence at Texas A&M University was an endless source of inspiration. He is the one who taught me a great deal about analog circuit design, and first introduced me to the exciting field of high-speed analog to digital conversion.

Tragically, Dr. VanPeteghem passed away in February of 1990, only a short time after initiating an ambitious research program involving the design of a high-speed converter. His disappearance came as a shock, and for a while threatened to stop the entire idea. Fortunately, thanks to the invaluable help of Dr. Randall Geiger and the Department of Electrical Engineering, it has been possible to carry on with this work. We know it would have been much different, and undoubtedly even more valuable had he been around. However, what he initiated has not been given up, and this dissertation represents the accomplishment of at least some of the goals he had set forth.

As mentioned, I am deeply indebted to Dr. Randall Geiger. He agreed to become my graduate advisor after Dr. VanPeteghem's disappearance, despite a impressive amount of other commitments. Throughout this work, he has always been extremely supportive and helpful. None of this would have been possible without him.

I would also like to acknowledge the unconditional support of the Electrical Engineering Department at Texas A&M University, in particular from Dr. Joe Howze and Dr. C. Singh. Among other things, they allowed me to teach several challenging classes as an assistant lecturer. This was a great, fulfilling experience, from which I feel I learned at least as much as the students I taught. For the specifics of this task, I have been helped tremendously by Dr. John Fleming, who

was senior lecturer and whose experience I greatly respect.

I strongly wish to thank the people at Texas Instruments Inc. of Dallas, TX, for the generous financial support they have provided for this project, as well as the advice, the expertise and the manufacturing resources for the prototype converter chip. In particular, I am extremely grateful to Dr. Richard Hester, who has agreed to become a committee member and has dedicated a large amount of his personal time to helping me with this research. An equally important contribution was made by Mr. Ming Chiang, manager of the Mixed Signal Design Department at Texas Instruments, who has made available many resources for the realization and evaluation of the prototype, and has been providing continuous moral support. I am also grateful for the strong support of Mr. Kenneth Buss, manager of Power IC's, with whom I have greatly enjoyed working. Finally, I would like to acknowledge the precious help of many other TIers, like Jim Hellums and Bill Krenik for the evaluation of new circuit configurations, Michiel Dewit for answering many specific questions, Sami Kiriaki and Greg Warwar for their help in testing, Tim Snodgrass for management of the design database and Mark Courtney in the Legal Department for writing patent applications.

I wish to thank all the committee members for their help in formalizing this text, as well as the other graduate students I had the pleasure to work with at Texas A&M University. In particular, I am grateful to Joe Spalding and Palaksha Setty, who helped design some sections of the prototype converter described below.

And finally, in a special way, I wish to express my gratitude to my wife Paulina, who has supported me unconditionally throughout this busy period. Not only did she help assemble this text, she also had to spend many nights alone. After more than two wonderful years of marriage, she is still waiting for a formal honeymoon. It would probably not be an overstatement to say that she indirectly deserves a large part of the credit for this dissertation and what it represents.

TABLE OF CONTENTS

CHAPTER	Page
Volume I	
I ANALOG TO DIGITAL CONVERTER PRINCIPLES	1
1.1. General Introduction	1
1.2. Abstract	2
1.3. General Concepts	3
1.4. Ideal, Static Transfer Curve	6
1.5. Non-Ideal, Static Transfer Curve	9
1.6. Conclusion	16
II CONCEPTS IN ANALOG TO DIGITAL CONVERTER TESTING	17
2.1. Introduction	17
2.2. Histogram Analysis	17
2.3. Single-Sweep Histogram Test	24
2.4. Distortion of the Ramp Waveform	29
2.5. Histogram Analysis Using Sine Waves	32
2.6. Noise in a Histogram Test	35
2.7. Combining Histograms	38
2.8. Harmonically Related Input Signals	40
2.9. Fourier Analysis	43
2.10. Conclusion	47
III DYNAMIC HISTOGRAM TESTING	49
3.1. Introduction	49
3.2. Dynamic Errors	49
3.3. Dynamic Histogram Test	57
3.4. Full Characterization	66
3.5. Conclusion	72
IV ANALOG TO DIGITAL CONVERTER ARCHITECTURES	73
4.1. Introduction	73
4.2. Operation of a Flash Converter	73
4.3. Multi-Step Converters	75
4.4. Pipelined Converters	80
4.5. Successive Approximation Converters	82
4.6. Recycling Converters	84

TABLE OF CONTENTS (Continued)

CHAPTER	Page
4.7. Time-Interleaved Converters	87
4.8. Oversampling Converters	91
4.9. Conclusion	99
V ERROR MODELING IN MULTI-STEP CONVERTERS	100
5.1. Introduction	100
5.2. Origin of the Errors	100
5.3. Error Analysis Methodology	104
5.4. DAC Errors	111
5.5. ADC Errors	113
5.6. Gain Errors	113
5.7. Combined Effect of Errors	116
5.8. Error Superposition Theorem	118
5.9. Derivation of Per-Stage Errors	120
5.10. Effect of Errors on Overall Integral Non-Linearity	122
5.11. Measurement of the Limit Histogram	123
5.12. Analysis of Errors in Subsequent Stages	129
5.13. Digital Error Correction	132
5.14. Dynamic Errors	135
5.15. Conclusion	138
VI DIGITAL ERROR CORRECTION IN PIPELINED CONVERTERS	139
6.1. Introduction	139
6.2. Generalized Pipelined Converter	139
6.3. Redundancy	149
6.4. Calibration Procedures	151
6.5. Estimation of the DAC Weights	151
6.6. Truncation Error	152
6.7. Monotonicity	155
6.8. Dimensioning of the Interstage Gains	156
6.9. Extension for Non-Linear Gain	156
6.10. High-Speed Operation	163
6.11. Noise	163
6.12. Advantages of Fully Digital Correction	165
6.13. ASIC Aspects	166
6.14. Conclusion	167

TABLE OF CONTENTS (Continued)

CHAPTER	Page
VII ACCURACY BOOTSTRAPPING	168
7.1. Introduction	168
7.2. General Principles	168
7.3. System Requirements	169
7.4. Calibration Procedure	177
7.5. Convergence of the Procedure	178
7.6. Conclusion	181
VIII AN ACCURACY-BOOTSTRAPPED A/D CONVERTER	182
8.1. Introduction	182
8.2. Basic Stage	182
8.3. Component Errors	186
8.4. Pipelined Converter	187
8.5. Control Hardware	192
8.6. Accuracy Bootstrapping Procedure	194
8.7. Error Tolerance	200
8.8. Nominal Code Patterns During Calibration	201
8.9. Linearity of the Uncalibrated Pipeline	202
8.10. Linearity of the Differential Measurements	204
8.11. Mappings and Coefficients	205
8.12. General Linearity Criterion	206
8.13. Experimental Results	210
8.14. Other Configurations	211
8.15. Intuitive Explanation of Convergence	214
8.16. Recycling Converters	215
8.17. Amplifier Non-Linearity	216
8.18. Practical Implementation	216
8.19. Conclusion	217
Volume II	
IX ANALYSIS OF SWITCHED-CAPACITOR GAIN STAGES	218
9.1. Introduction	218
9.2. Basic Operation	218
9.3. Gain and Speed	223
9.4. Accuracy	227

TABLE OF CONTENTS (Continued)

CHAPTER	Page
9.5. Slew Rate	229
9.6. Linearity	229
9.7. Switching Speed	233
9.8. Noise	235
9.9. Clock Feed-Through	236
9.10. Stability	239
9.11. Design Methodology	242
9.12. Conclusion	243
X THE CHARGE AMPLIFIER	245
10.1. Introduction	245
10.2. Basic Operation	245
10.3. Clocking Scheme	246
10.4. Analysis of the Precharge Phase	250
10.5. Analysis of the Initial Discharge Phase	250
10.6. Analysis of the Load Phase	255
10.7. Analysis of the Flash Phase	257
10.8. Analysis of the DAC Phase	263
10.9. Analysis of the Final Discharge Phase	270
10.10. Transient Waveforms	272
10.11. Repeatability of the Switching	276
10.12. Accommodation of Lead and Trail Times	281
10.13. Mismatches in Transient Currents	282
10.14. Multiplexing Comparator References	282
10.15. Summing Nodes	284
10.16. Input and Output Stages	284
10.17. Linearity	289
10.18. Implementation of the Current Sources	291
10.19. Channel Charge Redistribution	302
10.20. Power Consumption	302
10.21. Speed Advantage Over an SC stage	304
10.22. Noise Performance	305
10.23. Supply Noise Rejection	305
10.24. Conclusion	307

TABLE OF CONTENTS (Continued)

CHAPTER	Page
XI GENERAL NOISE THEORY	309
11.1. Introduction	309
11.2. Switched Versus Constant-Bias Systems	309
11.3. General Noise Concepts and Assumptions	312
11.4. Fourier Transforms	315
11.5. Parseval's Theorem and Energy Spectra	317
11.6. Autocorrelation Functions and Power Spectra	319
11.7. Expected Power	320
11.8. White and Colored Noise	321
11.9. Fourier Coefficients, Energy and Power Spectra	323
11.10. Autocorrelation of Fourier Coefficients	323
11.11. Power Spectra and Expected Power	326
11.12. Conclusion	326
XII NOISE IN SWITCHED SYSTEMS	328
12.1. Introduction	328
12.2. Hypotheses and Assumptions	328
12.3. Noise Statistics in Switched Systems	331
12.4. Expected Energy Spectrum of a Restricted Process	333
12.5. Phase Considerations	334
12.6. Odd-Symmetry Random Process	335
12.7. Expected Noise Power in Switched Systems	344
12.8. Conclusion	346
XIII CORRELATED DOUBLE SAMPLING AND NOISE	349
13.1. Introduction	349
13.2. Correlated Double Sampling in the Charge Amplifier	349
13.3. Principles of Correlated Double Sampling	349
13.4. Noise Cancellation	353
13.5. Correlated Double Sampling and Switched Noise	355
13.6. Conclusion	358
XIV NOISE IN THE CHARGE AMPLIFIER	359
14.1. Introduction	359
14.2. Charge Amplifier Components	359
14.3. Noise Sources	363

TABLE OF CONTENTS (Continued)

CHAPTER	Page
14.4. Noise During the Precharge Phase	366
14.5. Noise During the Initial Discharge Phase	367
14.6. First Switching Model: Stable Feed-Back	372
14.7. Second Switching Model: Sequential Switching	380
14.8. Ring Oscillator Analogy	385
14.9. Noise During the Load Phase	385
14.10. Noise During the Final Discharge Phase	390
14.11. Simulated Noise Performance	392
14.12. Supply Noise Rejection	400
14.13. Conclusion	402
XV IMPLEMENTATION OF THE PROTOTYPE CONVERTER	403
15.1. Introduction	403
15.2. Clock Generation	404
15.3. Basic Pipeline Stage	408
15.4. Current Sources and DAC Section	413
15.5. Offset Compensation in the Flash Section	415
15.6. Data Latches	415
15.7. Input Stage	417
15.8. In-Stage Calibration Circuitry	419
15.9. Calibration Logic	421
15.10. Reset Circuitry	428
15.11. Output Multiplexing	429
15.12. Data Capture and Microcomputer Support	429
15.13. On-Chip Digital Error Correction	431
15.14. Recycling Converters	433
15.15. Time-Interleaving	433
15.16. Conclusion	436
XVI CONCLUSION	437
16.1. Scope	437
16.2. Summary	437
16.3. Suggested Further Research	441
REFERENCES	443
APPENDIX: ACCURACY BOOTSTRAPPING PROGRAM	452
VITA	457

LIST OF FIGURES

FIGURE	Page
Volume I	
1. Ideal Transfer Curve	5
2. Quantization Error	8
3. Non-Ideal Transfer Curve	10
4. Quantization Error of a Non-Ideal Converter	12
5. INL and Quantization Error	14
6. Differential Non-Linearity	15
7. INL Error	22
8. Ramp Waveforms	25
9. Single Sweep	26
10. Ramp Response	31
11. Histogram with U-Shaped Bias	33
12. Probability Density Function Affected by Noise	37
13. Combining Two Histograms	39
14. Harmonically Related Input Signal	41
15. Histogram of Harmonically Related Waveform	42
16. Unfinished Period	44
17. Typical FFT Response	46
18. Signal Locus in the Magnitude-Slope Plane	52
19. Locus of a Ramp	54
20. Locus of an Asymmetric Ramp	55
21. Ellipsoidal Locus	56
22. Relationship between Samples and Locus	58
23. Continuous and Discrete Histogram Domain	60
24. Three-Dimensional Histogram Plot	61
25. Discrete Locus with More Than One Slope per Magnitude	62
26. One to One Mapping	63

LIST OF FIGURES (Continued)

FIGURE	Page
27. DNL Values along a Horizontal Line	68
28. Transition Levels along a Horizontal Line	70
29. Magnitude-Dependent Timing Error	71
30. Flash Converter	74
31. Multi-Step Converter	76
32. Sub-Ranges	77
33. Cascaded Converter Stages	79
34. Pipelined A/D Converter	81
35. Successive Approximation Converter	83
36. Unfolded Successive Approximation Converter	85
37. Recycling Converter	86
38. Partially Recycling (Hybrid) Converter	88
39. Timing of a Recycling Converter	89
40. Timing of a Pipelined Converter	90
41. Time-Interleaved Pipelined Converter	92
42. Basic Sigma-Delta Converter	95
43. Equivalent Sigma-Delta System	96
44. Model with Quantization Noise	97
45. Basic Multi-Stage Scheme	102
46. Residue Calculation	103
47. Residue Function of Ideal Stage	105
48. Residue, after Gain Stage	106
49. Probability Density Functions	108
50. Limit DNL Curve	109
51. Limit INL Curve	110
52. Stage with a DAC Error	112

LIST OF FIGURES (Continued)

FIGURE	Page
53. Stage with a Flash Error	114
54. Stage with Excessive Gain	115
55. Stage with Insufficient Gain	117
56. Stage with DAC and Flash Error	119
57. Typical 3-bit INL Curve	124
58. Peak and Gap in a Typical Histogram	126
59. Smoothing Effect of Noise	127
60. Histogram Affected by Non-Ideal Subsequent Stages	128
61. Bin with Two Peaks	130
62. Measured Histogram of a Commercial Converter	131
63. Measured INL Curve	133
64. Reconstructed INL Curve	134
65. Stage with Flash Error and Over-Range	136
66. INL Curve with Error Correction	137
67. Schematic of a Pipeline Stage	141
68. Flash Converter	142
69. Stage with Switched-Capacitor S/H	143
70. Symbolic Representation of a Stage	144
71. Cascaded ADC Stages	145
72. Stage with Pipelined Arithmetic Section	153
73. Three Stages with Arithmetic Section	154
74. Non-Linear Gain Characteristic	157
75. Piece-Wise Linear Approximation	158
76. PWL Approximation Error	160
77. System Consisting of Reconfigurable Blocks	171
78. Different Configuration of the System	172

LIST OF FIGURES (Continued)

FIGURE	Page
79. Reconfigurable Pipelined A/D Converter	173
80. System with Configuration Control Capability	174
81. Conceptual Schematic of One Block	176
82. Accuracy Bootstrapping Algorithm	179
83. Application to a 3-Stage Pipelined Converter	180
84. Schematic of One Pipelined Stage	183
85. Ideal Transfer Curve of the Stage (Residue)	185
86. Converter Stage with Digital Correction	189
87. Schematic of the Pipeline	190
88. Pipeline with Error Correction	191
89. Calibration Circuitry	195
90. Feed-Back Configuration	197
Volume II	
91. Non-Inverting Switched-Capacitor Stage	220
92. Inverting Switched-Capacitor Stage	221
93. Simplified Model of the Stage	224
94. Linear Gain	231
95. Non-Linear Gain	232
96. Linearized Transfer Curve	234
97. Effect of Clock Feed-Through	237
98. Bottom-Plate Switching	240
99. Basic Charge Amplifier Components	247
100. Basic Capacitor Discharge Waveform	248
101. Precharge Phase with Switch	251
102. Initial Discharge Phase and Offset Cancellation	253
103. Load Phase	256
104. A/D Converter Stage	258

LIST OF FIGURES (Continued)

FIGURE	Page
105. Detail of the A/D Converter Stage	259
106. Cancellation of Undershoot on the Reference Levels	261
107. Detail of Undershoot Cancellation	262
108. Undershoot Cancellation Using Capacitive Divider	264
109. A/D Converter Stage, Main Components	265
110. A/D Converter Stage (Detailed Schematic)	266
111. DAC, Using Auxiliary Current Sources	267
112. Generation of the Timing Pulse	269
113. Realization of the Charge Gain	271
114. Conventional Current Mirror	273
115. OFF-ON Current Transient	275
116. ON-OFF Current Transient	277
117. Combination of Comparator Output and Clock Pulse	279
118. Multiplexing of Comparator References (CMOS)	283
119. Multiplexing of Comparator References (Bipolar)	285
120. Input Stage	287
121. Alternative Input Stage	288
122. Output Stage	290
123. Triple Cascode Current Source	293
124. Regulated Cascode Current Source	294
125. Output Current Response (AC)	297
126. Output Current Response (Step)	298
127. Capacitor Voltage Response (AC)	299
128. Capacitor Voltage Response (Step)	300
129. Voltage Response, Gate of M_3 (AC)	301
130. Voltage Response, Gate of M_3 (Step)	303

LIST OF FIGURES (Continued)

FIGURE	Page
131. Simple Comparator Scheme	306
132. Sample of a Stochastic Process	314
133. Autocorrelation of Band-Limited Noise	322
134. Probability Distribution of Fourier Coefficients	324
135. Typical 1/f Noise Bursts	332
136. Half Processes	337
137. Finite Process Example	338
138. Combinations of Samples in the Finite Process	339
139. Positive Half Samples	340
140. Circular Symmetry of Fourier Coefficients	342
141. Cross-Section of Fourier Coefficients	343
142. Multiplying Function	347
143. Concept of Correlated Double Sampling	351
144. Timing of the Samples	352
145. Charge Amplifier Components	360
146. Regulated Cascode	362
147. Reference Voltage Multiplexer	364
148. Capacitor Voltage	369
149. Capacitor Charge Current	370
150. Capacitor Charge Current in Phase 2	371
151. Comparator Schematic	373
152. Time Jitter	374
153. Switch-OFF Current Transient	376
154. I/V Characteristic of the Transconductor	377
155. Equivalent Model of the Feed-Back Network	379
156. Simulated Transients at Switch-Off	381

LIST OF FIGURES (Continued)

FIGURE	Page
157. Time Jitter Calculation	384
158. Ring Oscillator	386
159. Output-Referred Comparator Noise	394
160. Output-Referred Inverter Noise	396
161. Output-Referred Current Noise	397
162. Output-Referred Steady-State Current Noise	399
163. Typical Pipeline Configuration	401
164. Clock Timing	405
165. Two-Phase Clock Generator	406
166. State Machine	407
167. Six-Phase Non-Overlapping Clock Generation	409
168. Block Diagram of a Converter Stage	410
169. Basic Gain Stage with Flash Section	411
170. Converter Stage with DAC Section	414
171. Schematic of a Data Latch	416
172. Input Section of the Converter	418
173. Folded Cascode Amplifier	420
174. In-Stage Calibration Circuitry	422
175. Calibration Logic	424
176. Calibration Sequence	427
177. Output Multiplexer	430
178. On-Chip Calibration Hardware	432
179. Prototype Converter	434

CHAPTER I

ANALOG TO DIGITAL CONVERTER PRINCIPLES

1.1. General Introduction

This dissertation covers a wide range of concepts related to analog to digital converter characteristics, testing, architectures and building blocks. It does so from a strict theoretical point of view, as well as from a practical perspective. In that sense, it was intended as an A through Z guide to analog to digital converter design. Throughout the preliminary research, the need for such work was felt. Many people are very active in this field, but relatively little fundamental literature seems to exist on the subject. Due to the huge scope of the field, the discussion was mainly focused on pipelined architectures, although many concepts can be extrapolated to other schemes. An attempt has been made to include as many innovative ideas as possible.

Chapter I describes and defines general converter characteristics, like the static transfer curve, differential and integral non-linearity. Its value is mainly to define concepts that are almost universally used, but rarely defined. Chapter II describes several testing and evaluation methods, with a particular focus on histogram testing. This kind of testing has been known for a long time, but is not used as much as it could be, possibly because of a lack of understanding about the capabilities of the method. Chapter III is an attempt to extend the histogram concept, which has mainly been used for static testing, to include dynamic (high-frequency) effects. The key was to define the signal space as a two-dimensional entity, representing the magnitude as well as the first derivative of the input signal.

Chapter IV gives a general overview of converter architectures, with particular focus on pipelined, recycling and time-interleaved approaches. Chapter V is a detailed analysis of static error mechanisms in multi-step and pipelined converters. Chapter VI takes the same concepts a step further, and provides a method to digitally correct all these errors. Chapter VII introduces a novel self-calibration method, called accuracy bootstrapping. The method makes it possible

to efficiently identify errors within complex systems designed to realize a very precise transfer function. Chapter VIII applies the general concept of accuracy bootstrapping to a pipelined converter scheme.

Chapter IX mathematically describes switched-capacitor sample/hold amplifiers, one of the key elements of pipelined converters. It is shown that this conventional approach has some shortcomings. Chapter X introduces a novel, alternative sample/hold amplifier architecture, called the charge amplifier. It is expected that this scheme could provide superior performance and lowered power consumption.

The charge amplifier has a very particular noise behavior, since it is a switched system. Chapter XI summarizes a number of important noise concepts, needed in order to perform a mathematical analysis of noise in the charge amplifier. Chapter XII investigates from a purely theoretical point of view how some of these concepts should be interpreted in order to model noise in the particular case of switched systems. Chapter XIII introduces correlated double sampling, which is used in the charge amplifier, and its ability to reduce offsets and low-frequency noise components. Chapter XIV is the actual noise analysis of the charge amplifier, based on formulas derived in the previous chapters.

Finally, chapter XV describes an actual prototype pipelined analog to digital converter, based on the charge amplifier and self-calibrated with accuracy bootstrapping. According to theoretical results and circuit simulations, its performance should match or exceed that of current state of the art devices.

1.2. Abstract

The remainder of this chapter starts with a general definition of analog to digital converters. The definition is gradually made more specific, and the concept of static transfer curve is introduced. From the transfer curve, several key qualifiers of static (DC) converter performance are derived, like integral and differential linearity. Although these notions are well-known, it was found useful to define them in strict mathematical terms, since literature about this subject is scarce and some misconceptions exist.

The definitions of this chapter are a compilation, as well as an extension of concepts introduced or discussed in a scattered way in earlier manufacturers'

product notes (e.g. Analog Devices [1] and Hewlett Packard [2]) and publications by authors like Tewksbury [3], Doernberg [4], Max [5], the IEEE Waveform Measurement and Analysis Committee [6] and more recently by Geiger [7]. These definitions will be the basis for further analysis of specific converter types. In particular, digital error-correcting techniques described in later chapters will extensively rely on them.

1.3. General Concepts

An analog/digital converter (ADC) is a device that takes an analog input signal and produces a corresponding digital output code. The input signal is usually represented by a voltage, but could be a current, or a charge stored on a capacitor as well. In some cases, like the specific prototype that will be described in this dissertation, the input can even be a pulse-width modulated digital signal.

In either case, a fundamental aspect of an ADC is that the input signal is continuous, while the output signal is discretized. The converter is always implemented in a way that limits the allowable values of the input signal to a certain range of voltage, current or pulse width. Beyond that range, R , the conversion process fails due to inherent limitations of the electronics. Because of its nature, the input signal must be represented by a real number, within a certain numerical range. The value can have units of volts, amps, coulombs or seconds, although in abstract discussions about ADC's, the input signal will often be represented in a normalized way with respect to the input range. Depending on the situation, it may carry a value between 0 and 1, or within a numerical range that corresponds to the range of possible binary numbers at the output.

The range of output signals is naturally limited. The output normally consists of a number of digital output lines (bits), which together represent some digital word. Obviously, the very number of output bits limits the number of possible, distinct values of the output signal. If there are N output bits, the number of possible, discrete output combinations is 2^N . The number of output bits is also called the resolution of the converter. It will be shown later that the resolution is not a sufficient measure of the converter's performance by itself.

An ADC defines a mapping between some input signal values and an output code. Obviously, this mapping cannot be one-to-one, since there is an infinite

number of possible analog input values within the range R , and a finite number of possible output combinations. However, the mapping must satisfy the definition of a function: to each possible value at the input must correspond one and exactly one output code. This function is called the transfer function of the system. The *accuracy* (predictability) of the transfer function is extremely important, and truly reflects converter quality, rather than the resolution (number of output bits). Devices that do not satisfy the function requirement are subject to random effects (noise), memory effects or hysteresis. Depending on the relative magnitude of these effects with respect to the desired accuracy of the transfer function, a device can be worthless as a digitizer.

The mapping between input and output can occur in different ways. Probably most common is the linear mapping, in which different output codes are assigned to nominally equal sections of the input range. Some applications, like telecommunications (voice coding) use logarithmic or semi-logarithmic mappings, but those will be largely ignored throughout this discussion. Linear mappings normally assign the output codes so that to input range segments of increasing magnitude would correspond output codes of increasing or decreasing binary value. This is called monotonicity. Depending on whether straight binary or two's complement arithmetic conventions are used, the output can be considered unipolar or bipolar. The latter case is obviously used to code signals that can be either positive or negative.

Throughout this work, we will focus exclusively on ADC's that aim at implementing linear, monotonous and unipolar mappings between the input signal and the output code. In particular, we will assume that an output code with binary equivalent of 0 corresponds to the lower end of the input range, while the binary equivalent of $2^N - 1$ corresponds to the higher end. The ideal transfer function of such an ADC will map the continuous input range (mathematically modeled as a subset of real numbers) to a finite, discrete range of integers. The shape of the ideal transfer curve is shown on figure 1, for the simplified case of a converter with 3 bit resolution (8 possible output codes). The transfer characteristic is called ideal (or linear), because a straight line can be drawn through the breakpoints of the curve.

The larger part of the research described in this dissertation has been devoted

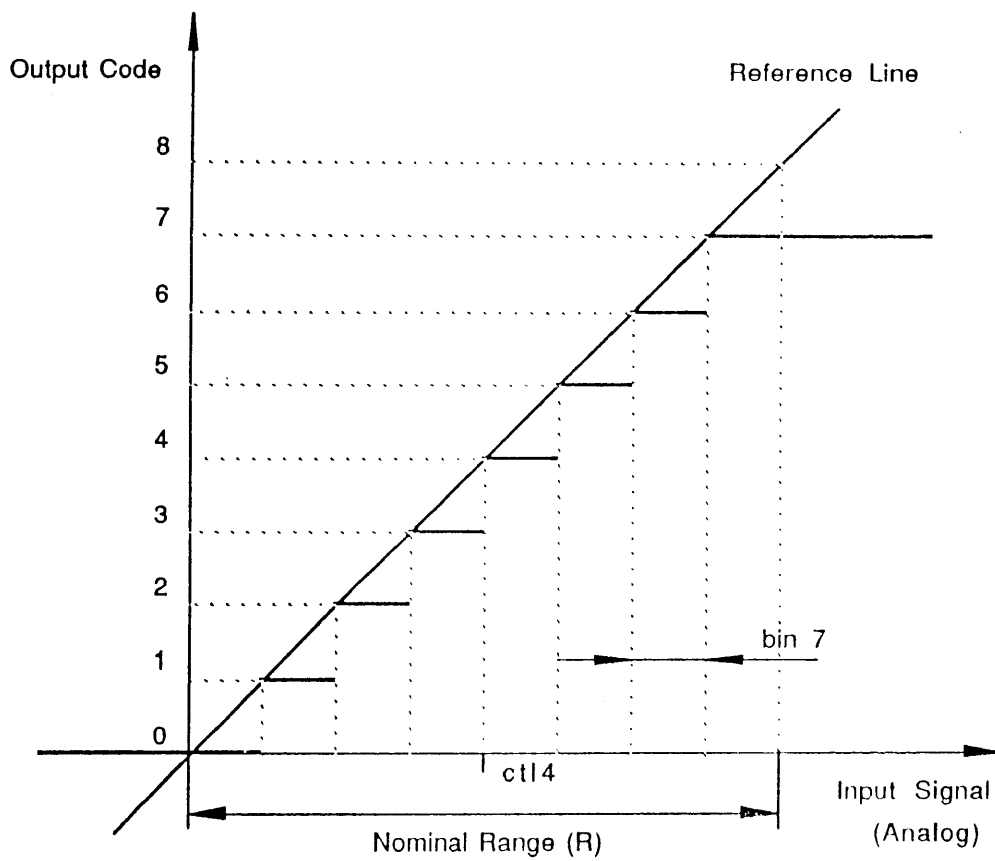


Fig. 1: Ideal Transfer Curve

to trying to linearize the transfer curve of a specific type of ADC's: pipelined ADC's. In order to achieve this goal, a thorough understanding first had to be gained about the operation of a pipelined converter. The fundamental effects that limit its performance (being the ability to implement a transfer function close to the ideal one) had to be investigated. Finally, new circuit techniques were developed in an attempt to circumvent the influence of imperfect or unpredictable circuit elements upon the transfer curve.

The scope of the problem could be described as follows: given the fact that electronic devices on an integrated circuit have tolerances in the range of 0.1% to several %, how do we build a reliable pipelined converter with an accuracy better than 12 bits, or 1/4096 (0.025%)?

1.4. Ideal, Static Transfer Curve

Different terminology has been used by several authors for referencing the transfer curve. We will maintain the following conventions. The input range R is a subset of all possible analog input values, chosen conventionally so that within that range, a "reasonably" linear ADC transfer curve is obtained.

We will consider a certain number of output bits N , and we will assume that any one of their 2^N possible binary combinations (output codes) corresponds to a specific subsection of the input range. Ideally, the input range is divided into 2^N identical subsections. If R^- is the lowest value within the range and R^+ is the highest value, each subsection will ideally have a width of $(R^+ - R^-)/2^N$. Each of these subsections will be referred to as a *bin*. Their width is called the ideal bin width W_{id} .

$$W_{id} = \frac{R^+ - R^-}{2^N} = \frac{R}{2^N} \quad (1)$$

The bins will be numbered from 0 to $2^N - 1$, according to their corresponding binary output value. For the ideal ADC we will consider, an output value of 0 corresponds to the lowest bin and consecutive integers correspond to subsequent bins. The possible input values x of bin i are such that

$$R^- + i W_{id} \leq x < R^- + (i + 1)W_{id} \quad (2)$$

The smallest possible analog value in bin i , x_i^- , is given by

$$x_i^- = R^- + i W_{id} \quad (3)$$

Since i is the bin number as well as the output code corresponding to bin i , a straight line can be drawn on the diagram showing the ideal transfer curve, through all the points with coordinates (x_i^-, i) . The slope of this line is $2^N/R$, and it will be called the reference line. The reference line is tangent to the ADC transfer curve for input values x_i^- , as defined above. For other input values, the transfer curve lies below the reference line.

The difference between the reference line and the transfer curve is called the quantization error. It can be considered the inherent error of the conversion process. As a function of the input signal, the quantization error has a sawtooth shape with values between 0 and 1 (figure 2). The input-referred quantization error has values between 0 and W_{id} , and is sometimes called the residue of the conversion. Since $W_{id} = R/2^N$, it is clear that the maximum input-referred quantization error of an ideal converter can only be reduced by increasing N .

It is not always convenient to refer to the input signal in terms of voltage, current or any other analog quantity. Instead, the signals are often normalized with respect to the input range (linear scaling), so that a value of 0 would correspond to R^- and a value of 2^N to R^+ . The new units in that case are called *lsb* (for least significant bit). As a result, $W_{id} = 1 \text{ lsb}$, which is also the maximum residue of an ideal converter.

In some cases, we may want to consider input signals outside of the conventional input range. For that purpose, we will define two additional bins, bin -1 and bin 2^N . Bin -1 contains all inputs below R^- , bin 2^N contains all inputs equal to or greater than R^+ . Bin -1 will also be referred to as the underflow bin, bin 2^N as the overflow bin. As opposed to the other bins, the underflow and overflow bins are ideally infinite. Although no ADC output code is defined for these bins, in many converter schemes a binary value of 0 corresponds to the underflow bin and a value of $2^N - 1$ to the overflow bin. In other words, the underflow bin is an extension of bin 0, and the overflow bin an extension of bin $2^N - 1$. Sometimes, one or two additional outputs are provided to indicate an over-or underflow condition.

The input signal which defines the transition between bin $i - 1$ and bin i (and the corresponding transition in output code) will be called the *ideal code transition level i* , or $ctl_{i,id}$. By definition, $ctl_{0,id} = R^-$ and $ctl_{2^N,id} = R^+$. Obviously, for

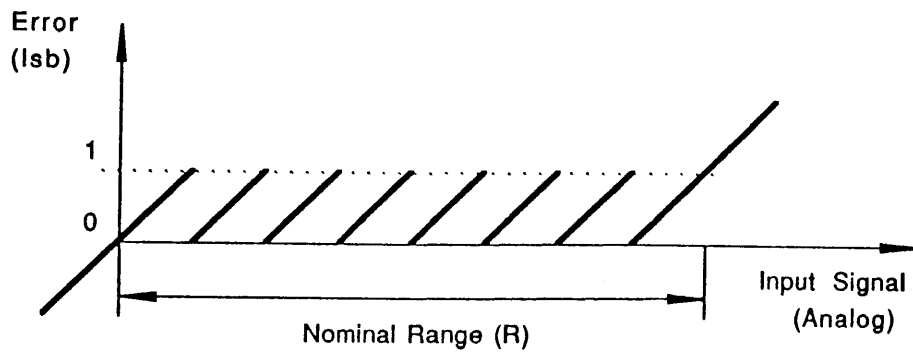


Fig. 2: Quantization Error

any i so that $0 \leq i < 2^N - 1$, the following properties apply

$$W_{id} = ctl_{i+1} - ctl_i = 1 \text{ lsb} \quad , \text{ and } \quad ctl_{i,id} = i \text{ lsb} \quad (4)$$

1.5. Non-Ideal, Static Transfer Curve

Figure 3 shows the transfer curve of a non-ideal ADC. The curve is still a discontinuous function from a certain analog input range to a subset of integer numbers. However, a straight line cannot be drawn through the breakpoints of the transfer curve anymore. The ADC is said to exhibit non-linear behavior (to be distinguished from the intrinsic non-linearity expressed by the quantization error).

Most definitions introduced with the ideal ADC transfer curve can be extended to the non-ideal case. The nominal input range is still the range of values for which the ADC will exhibit reasonably linear behavior. However, it will be seen that precisely determining this range is not straightforward. A bin is defined as the set of input voltages yielding the same output code. Over-and underrange bin are defined as before; they contain input values outside of the nominal range. A non-ideal code transition level is the input value defining the transition between one output code (or the corresponding bin) and the next one. Finally, one lsb is defined as an analog unit corresponding to the width of the total analog input range, divided by 2^N .

The most straightforward way to describe the non-ideal ADC curve is by listing all its code transition levels. $2^N - 1$ of these levels, ctl_1 through ctl_{2^N-1} can easily be determined through a set of DC measurements. It is sufficient to slowly ramp up the input signal to the ADC, and to note for which input values the output flips. However, unless over-and underrange indication is provided by the ADC, this measurement will not reveal ctl_0 or ctl_{2^N} . These two levels, which determine the actual input range of the ADC under consideration, have to be determined conventionally. This can be achieved by assuming that bin 0 and bin $2^N - 1$ are ideal with respect to their bin width. The ideal bin width can be determined according to the formula

$$W_{id} = \frac{ctl_{2^N-1} - ctl_1}{2^N - 2} \quad (5)$$

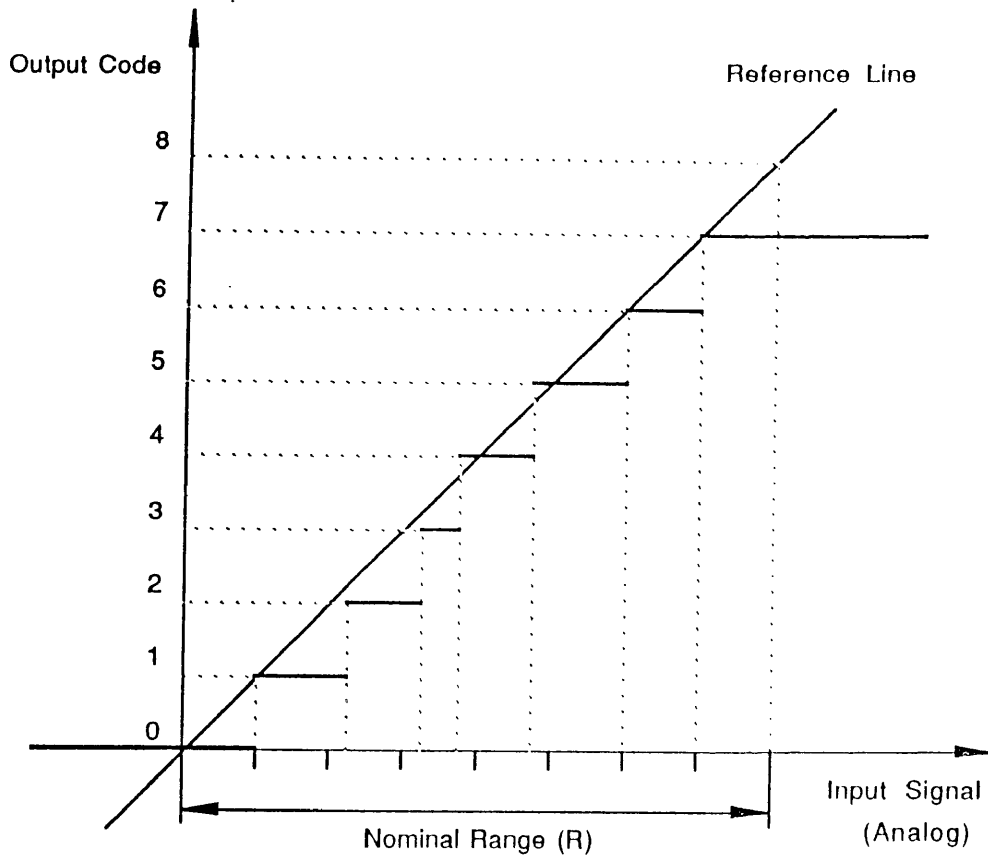


Fig. 3: Non-Ideal Transfer Curve

In that case

$$ctl_0 = R^- = ctl_1 - W_{id} \quad \text{and} \quad ctl_{2^N} = R^+ = ctl_{2^N-1} + W_{id} \quad (6)$$

The input range is then defined so that

$$R = ctl_{2^N} - ctl_0 = 2^N W_{id} = 2^N lsb \quad (7)$$

Once the input range is defined, the reference line of the non-ideal ADC can be drawn. By convention, we will draw the line through the two endpoints of the transfer curve, $(R^-, 0)$ and $(R^+, 2^N)$. Other conventions are possible and indeed used. For instance, some people determine the reference line through a best-fit method. The result of our definitions is that the points $(R^- + W_{id}, 1)$ and $(R^+ - W_{id}, 2^N - 1)$ always all lie on the reference line as well.

The quantization error can be defined like in the ideal case, as the difference between the reference line and the transfer curve. The quantization error of a non-ideal ADC does not necessarily lie between 0 and 1 *lsb* anymore. Changes in code transition levels alter the sawtooth pattern (figure 4).

An alternative way to characterize the non-ideal transfer curve is through the concept of *integral non-linearity* (INL). The $2^N + 1$ INL values, inl_i , of the converter are defined as the difference between the actual code transition levels, and the ideal code transition levels, corresponding to an imaginary ideal ADC with identical input range.

$$ctl_{i,id} = R^- + i W_{id} \quad , \quad \text{for } i = 0 \cdots 2^N \quad (8)$$

$$inl_i = ctl_{i,id} - ctl_i \quad , \quad \text{for } i = 0 \cdots 2^N \quad (9)$$

As a result of earlier definitions,

$$inl_0 = 0 \quad , \quad inl_1 = 0 \quad , \quad inl_{2^N-1} = 0 \quad , \quad \text{and} \quad inl_{2^N} = 0 \quad (10)$$

INL is normally expressed in *lsb* units. It is a very important measure of static (DC) ADC performance, since it reflects the quantization error with

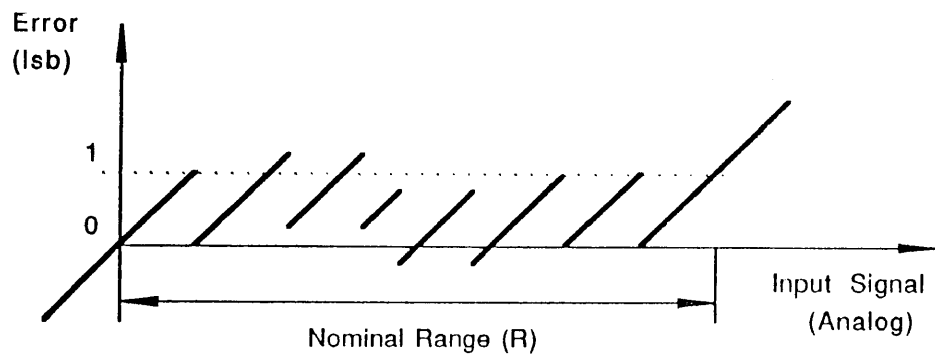


Fig. 4: Quantization Error of a Non-Ideal Converter

respect to the reference line. One can easily verify that the amount by which the quantization error exceeds 1 *lsb* or undershoots 0 *lsb*, is equal to the INL values. As such, INL expresses by how much the actual quantization error exceeds the maximum inherent error of the conversion process (figure 5). In addition, as will be shown later, knowledge of the individual INL values will allow us to determine all significant internal errors in the circuitry of a pipelined ADC.

In practice, an ADC is specified to be truly linear up to N bits if

$$|inl|_{max} \leq \frac{1}{2} lsb = \frac{1}{2} \frac{R}{2^N} \quad (11)$$

Due to imperfect design, ADC's with a physical output word length of N bits may only be linear to the n bit level, with the actual bit level n defined as

$$n = \log_2\left(\frac{R}{2 |inl|_{max}}\right) \quad (12)$$

Although in principle, n can be determined through the measurement of individual code transition levels, followed by the calculation of INL values, this procedure can be impractical. In particular, it requires that the transition levels be determined to an accuracy exceeding the maximum possible accuracy of the ADC under evaluation. Alternative methods exist, which allow very precise determination of INL values and transition levels, using only the ADC itself. One example is the histogram test, to be described below. This test relies on a slightly different description of the ADC transfer curve, called differential non-linearity (DNL).

DNL is defined as the difference between the ideal bin width of the converter and the actual width of each bin. The 2^N DNL values are usually expressed in *lsb* and given by (figure 6):

$$dnl_i = W_i - W_{i,id} = (ctl_{i+1} - ctl_i) - W_{i,id} \quad , \text{ for } i = 0 \cdots 2^N - 1 \quad (13)$$

As a result of the definitions, it is clear that

$$inl_i = \sum_{k=0}^i dnl_k \quad (14)$$

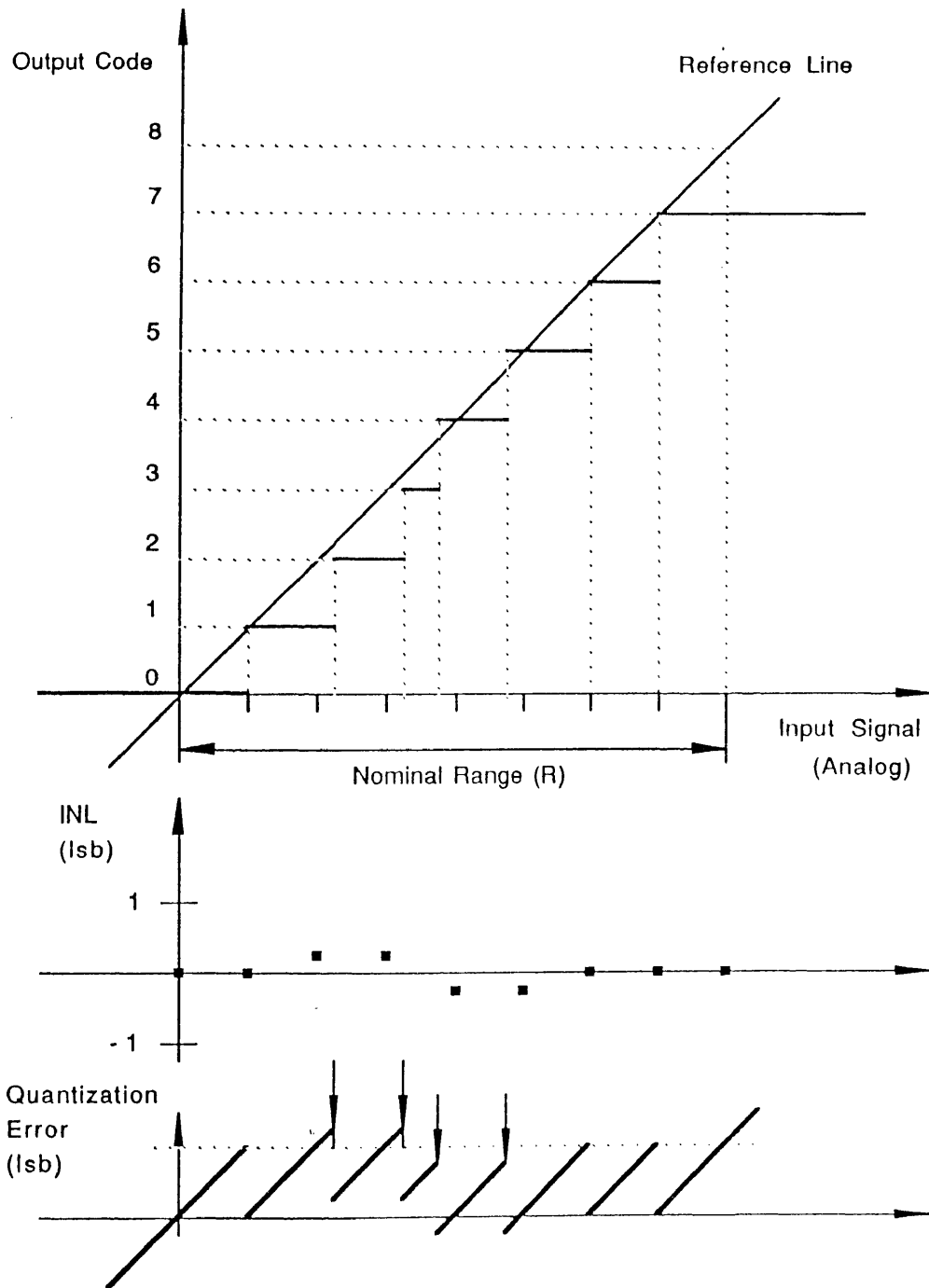


Fig. 5: INL and Quantization Error

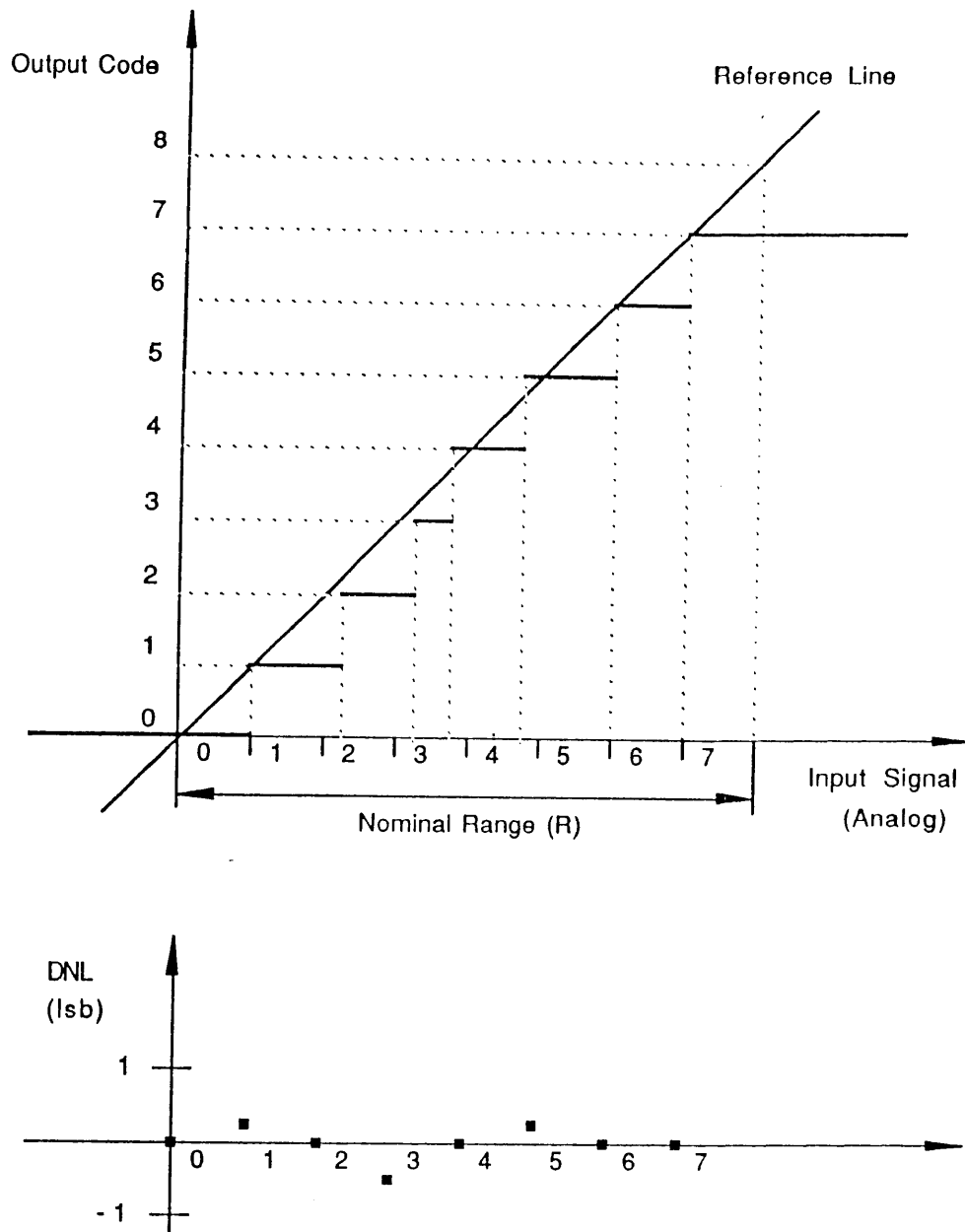


Fig. 6: Differential Non-Linearity

It is also clear that $dnl_0 = dnl_{2^N-1} = 0$ and that $dnl_i \geq -1 \text{ lsb}$, as long as the converter is monotonous ($ctl_{i+1} \geq ctl_i$).

As a result of the definitions we used, ADC gain and offset errors are excluded from the picture. This was found desirable in this case, since we mainly want to focus on overall linearity of ADC's. It also makes it easier to relate these concepts to the results of histogram tests or to distortion measurements, both of which are insensitive to gain and offset errors. The restriction is due to an implicit end-point normalization of reference line, DNL and INL curves, rather than a more complex mid-point or best-fit approach. It does not represent any loss in generality anyway, since offset and gain errors of the ADC can be determined independently by a two-point measurement, if estimated relevant.

1.6. Conclusion

In this chapter, a general definition of analog to digital converters was given. The definition was gradually refined, leading to the theoretical (ideal) transfer curve of linear converters. From this ideal curve, concepts like input range, transition level, quantization error, bin and bin width and *lsb* units were defined. By comparing the transfer curve of an actual converter to the ideal transfer curve, concepts like differential and integral non-linearity were introduced.

CHAPTER II

CONCEPTS IN ANALOG TO DIGITAL CONVERTER TESTING

2.1. Introduction

This chapter introduces some ADC testing methods, commonly used to determine the linearity of the ADC response to low-frequency (static) input signals. The concept of histogram testing [4,6] is derived using statistical methods, from previously discussed properties of the static ADC transfer curve. A great deal of attention is paid to the

estimation error associated with differential and integral non-linearity values derived through a histogram test. From these formulas, optimal test strategies are developed. In particular, it is investigated how the number of samples taken, the input waveform and the relationship between signal and sampling frequency affects the accuracy of a histogram test.

This rather detailed analysis was included because of the practical importance of histogram testing for the evaluation and diagnosis of

ADC's, and because existing literature is rather limited. Also, the concept of static histogram testing will be extended in chapter III, into a practical tool to perform dynamic (high-frequency), as well as static characterization. In chapter V, it will be seen how the ADC transfer curve obtained through histogram testing, can reveal and quantify the effect of many errors in the internal circuitry of an ADC [8]. These findings were fundamental to the development of high-performance ADC error correcting techniques described in subsequent chapters.

Finally, a short section is included about ADC testing using the fast Fourier transform (FFT) [4,6]. This method is used extensively in industry, because of its simplicity. However, it will be demonstrated that the FFT method does not reveal as much detailed information as a precision histogram test.

2.2. Histogram Analysis

The concepts of bin width and differential non-linearity lead to a practical, accurate statistical method to determine ADC non-linearity (be it in the form

of INL or of actual number of bits). The idea is the following. If one applies a random input signal to an ADC, the probability that a certain output code C will occur, is equal to the probability that the input signal was in bin C (since by definition bin C contains all possible input values yielding output code C).

We could now consider a random input signal of which the magnitude distribution or probability density function (the input signal is a continuous rather than discrete variable) is constant and equal to $1/R$ within the input range, and 0 otherwise. This means a random input signal that could have any value between R^- and R^+ , with equal probability for any value within that range. For an ideal converter, the probability that such a signal, x , would yield an output code C , is (with *pdf* the probability density function)

$$P(C) = \int_{R^- + (R/2^N)C}^{R^- + (R/2^N)(C+1)} pdf(x) dx = \frac{1}{R} \cdot \frac{R}{2^N} = \frac{1}{2^N} \quad (15)$$

For a non-ideal converter, this expression changes into

$$P(C) = \int_{ctl_C}^{ctl_{C+1}} pdf(x) dx = \frac{1}{R} (ctl_{C+1} - ctl_C) = \frac{1}{R} W_C \quad (16)$$

If R were known and $P(C)$ could be estimated for any C , the 2^N bin widths W_C of the converter could be calculated, and from there the DNL and INL values. One minor problem is that R cannot be easily determined, unless at least ctl_1 and ctl_{2^N-1} are measured accurately (using external precision instrumentation). However, the precise value of R does not need to be known to deduct from equation (16) that W_C is *proportional* to $P(C)$. This property is even maintained if *pdf*(x) were non-zero outside of the input range, as long as the pdf is constant within that range.

The probability of a certain output code, $P(C)$, can be estimated using established statistical methods. Since $P(C)$ is the probability that the input signal x would be in bin C , and since this fact can easily be established by verifying whether the corresponding output code is equal to C or not, the experimental determination of $P(C)$ assumes the form of a dichotomy. This is an experimental evaluation of the probability that a certain property would be

present in a population or not (a classical example of a dichotomy is an opinion poll to determine what percentage of a population will vote republican and what percentage democrat).

The actual value of $P(C)$ can be determined by taking a large number of samples with the ADC, while ensuring that the input signal maintains a uniform pdf over the input range. The fraction of samples for which the output code is C , is an approximation for $P(C)$.

$$P(C) \approx \frac{M_C}{M} \quad (17)$$

M is the total number of samples considered, M_C the number of samples yielding code C . The standard deviation σ on $P(C)$ can be estimated using the well-known dichotomy formula

$$\sigma \approx \sqrt{\frac{P(C) (1 - P(C))}{M}} \quad (18)$$

Both formulas yield reasonably accurate numerical values if the expected value of M_C is about 5 or more. In order to satisfy this condition for an approximately ideal N bit ADC, the total number of samples taken, M , must be greater than $5 \cdot 2^N$. If the ADC is known to be strongly non-ideal, and certain bins are significantly smaller than the ideal bin width, this number may have to be increased accordingly.

Of course, the experimental determination of $P(C)$ must be performed for any possible value of C . In practice, this is realized using a histogram test. A large number of samples ($M > 5 \cdot 2^N$) is taken, and the number of times each possible output code occurs, is counted (tallied). As demonstrated earlier, the ratios M_C/M will be approximately proportional to the bin widths W_C (within the accuracy limits set by the standard deviation, expressed in equation (18)).

A difficulty arises when one considers the first and the last bin of the ADC, with code 0 and $2^N - 1$ respectively. Unless the ADC has over-and underrange indications, or the input signal is made to coincide *exactly* with the input range, without exceeding it, the number of samples counted in these bins will be disproportionate and the information useless. In practice, this situation is avoided by considering only samples with output codes between 1 and $2^N - 2$. As a result,

no information will be available about the first and the last bin, but this does not represent any limitation, since we have conventionally set their bin widths equal to the ideal bin width W_{id} . If we call the total number of non-discarded samples M' , M_C/M' will now express $P(C)$. Obviously, the following relation will hold.

$$\sum_{C=1}^{2^N-2} P(C) = \sum_{C=1}^{2^N-2} \frac{M_C}{M'} = \frac{M'}{M'} = 1 \quad (19)$$

Since each $P(C)$ is proportional to W_C and $P(C)$ is proportional to M_C , each of the tallies M_C will be proportional to the respective bin widths W_C . In a similar way, $\sum M_C = M'$ will be proportional to $(ctl_{2^N-2} - ctl_1)$ (the input range minus the first and last bin). Since by definition, this reduced range has a width of $2^N - 2$ lsb , the proportionality constant k can be determined as

$$k = \frac{M'}{2^N - 2} lsb^{-1} \quad (20)$$

It is clear that the numerical value of k is also equal to the expected number of samples per bin (since the expected width of one

bin is 1 lsb). Once k has been determined, the actual width W_C of each bin can be determined. The value is again expressed in lsb , and as a result no absolute measurement of the input range is necessary. In other words: the histogram test makes it possible to determine the normalized bin widths (in lsb) of an ADC, using only the ADC itself.

$$W_C \approx \frac{M_C}{k} = \frac{M_C}{M'} (2^N - 2) lsb \approx P(C) (2^N - 2) lsb \quad (21)$$

In addition,

$$W_0 = W_{2^N} = 1lsb \quad (\text{definition}) \quad (22)$$

The standard deviation on the bin width is equal to $(2^N - 2)$ lsb times the standard deviation on $P(C)$, or according to equation (18):

$$\sigma_{W;C} = (2^N - 2) \sqrt{\frac{P(1-P)}{M'}} lsb = \frac{2^N - 2}{M'} \sqrt{M_C (M' - M_C)} lsb \quad (23)$$

For a close to ideal ADC, $M_C \approx M'/(2^N - 2)$, and the expression can be further simplified to

$$\sigma \approx \frac{2^N - 2}{M'} \sqrt{\frac{M'((2^N - 2)M' - M')}{(2^N - 2)^2 M'}} lsb = \sqrt{\frac{2^N - 3}{M'}} lsb \quad (24)$$

The minimum number of samples in order to determine W with a standard deviation of σlsb , is

$$M'_{min} = \frac{2^N - 3}{\sigma^2} \quad (25)$$

From the bin widths, differential and integral non-linearity can be determined (all expressed in lsb).

$$dnl_i = W_i - 1lsb \quad , \quad \text{and} \quad inl_i = \sum_{l=0}^i dnl_l = \left(\sum_{l=0}^i W_l \right) - i lsb \quad (26)$$

Equation (24) is fairly commonly applied to calculate the estimation error on the bin widths W_i (halving the standard deviation requires quadrupling of the number of samples). However, the same formula cannot be applied to estimate the accuracy of INL values. As a rough approximation, some people argue that since INL is calculated as a sum of DNL values, the error on the sum will be the error on one term, *multiplied* by the square root of the number of terms (which could be up to 2^N). This reasoning is not justified, since the different DNL values are not statistically independent. However, the approximation happens to predict the worst-case INL error reasonably well, like will be shown next.

Considering an INL value to be a sum of DNL values is one way to look at the problem. Another way is to consider inl_i as the error (referred to an ideal ADC) on the width of a segment of the input range which contains all possible values yielding output codes C between 0 and i , as shown on figure 7. Because we discard bin 0 (zero error, by definition), we could also consider the segment of values with corresponding output codes between 1 and i . The error on the width (in lsb) of this segment can be determined directly from the histogram tallies, using a reasoning identical to the one used to calculate the DNL values.

$$inl_i = \frac{\sum_{C=1}^i M_C}{k} - i lsb \quad (27)$$

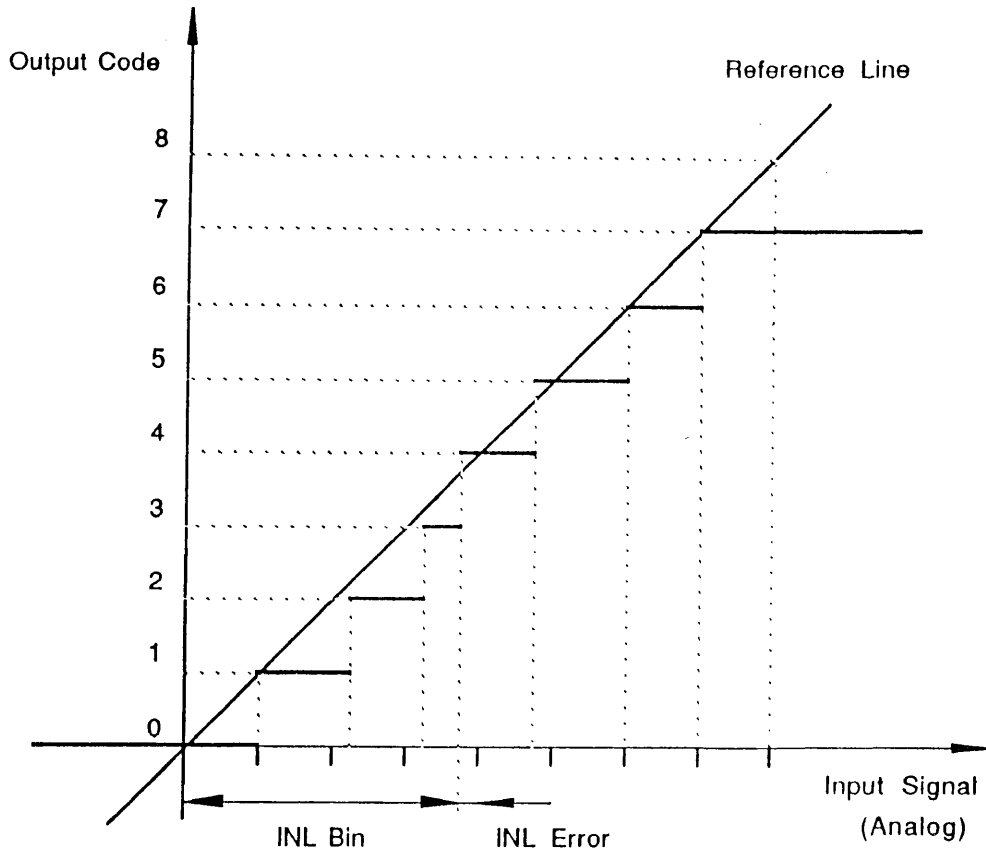


Fig. 7: INL Error

$$inl_i = (2^N - 2) lsb \frac{\sum_{C=1}^i M_C}{M'} - i lsb = (2^N - 2) lsb P(1 \leq C \leq i) - i lsb \quad (28)$$

The standard deviation on this estimate can be calculated according to equations (18) and (24):

$$\sigma(inl_i) = (2^N - 2) lsb \sigma(P(1 \leq C \leq i)) \quad (29)$$

$$\sigma(inl_i) \approx (2^N - 2) \sqrt{\frac{P(1 \leq C \leq i) P(i+1 \leq C \leq 2^N - 2)}{M'}} lsb \quad (30)$$

For a close to ideal ADC, this expression is approximately a quadratic function of i . $\sigma(inl_i)$ is 0 for $i = 0$ and $i = 2^N$ (definition), equal to the value of equation (24) for $i = 1$ and $i = 2^N - 1$ ($inl_1 = dnl_1$ and $inl_{2^N-1} = dnl_{2^N-1}$). It is maximized for $i = 2^{N-1}$, when $P(1 \leq C \leq i) \approx P(i+1 \leq C \leq 2^N - 2) \approx 1/2$. In that case, the standard deviation is given by

$$\sigma(inl_i) \approx (2^N - 2) lsb \sqrt{\frac{\frac{1}{2} \frac{1}{2}}{M'}} = \frac{2^{N-1} - 1}{\sqrt{M'}} lsb \quad (31)$$

The minimum number of samples in order to determine inl_{2^N-1} with an accuracy of σlsb , is

$$M'_{min} = \frac{(2^{N-1} - 1)^2}{\sigma^2} \approx \frac{2^{2N-2}}{\sigma^2} \quad (32)$$

This is significantly more (about $2^N/4$ times) than the number of samples required to determine the DNL values to the same accuracy. It is clear that for ADC's with outputs in excess of about 8 bits, the number of samples quickly becomes prohibitive. Fortunately, the histogram test is rarely performed with truly random signals, and the formulas derived above may be unnecessarily pessimistic.

In practice, it is almost impossible to generate a truly random signal with uniform probability density function. Instead, deterministic signals with uniform

voltage distribution are commonly used. Examples of such signals are the symmetric triangular wave or linear ramp, the asymmetric triangle and the sawtooth wave (figure 8).

Because their voltage densities are uniform (at least over a limited range), most properties of the histogram test will be maintained. However, the fact that these signal are deterministic rather than random has important effects upon the expected estimation error (σ) of the bin widths, DNL and INL values derived from the histogram.

2.3. Single-Sweep Histogram Test

Many people routinely perform histogram tests using periodic waveforms like described above. Determining the estimation error on the histogram under such conditions is not straightforward, since it depends heavily on the correlation between input signal frequency and ADC sampling frequency. However, one limit case lends itself very well to an error analysis, and can later be extended to more general situations. That case is a single, linear sweep of the input signal. The situation can be seen as a histogram test using a linear ramp as input, during only one half period of the ramp (figure 9).

The sweep consists of applying an input signal that increases linearly in time, such that all values within the input range of the ADC would be represented. The momentary value x of the sweep can be expressed as $x(t) = a t$. a is constant and represents the time derivative of the input signal, in units of input (V , A , $s\dots$) per unit of time (s). To keep this discussion general, we will normalize the input signal with respect to the ADC input range and express the time derivative of the sweep in units of lsb/s . Since by definition, the input range R is equal to $2^N lsb$, we can rewrite the sweep as

$$x(t) = b t \quad , \quad b = a \frac{2^N lsb}{R} \frac{1}{s} \quad (33)$$

We will assume that the ADC has a constant sampling rate f_s , expressed in samples per second or Hz . One can then easily calculate the average number of samples per lsb , or the expected number of samples per bin. k .

$$k = \frac{f_s}{b} \quad (34)$$

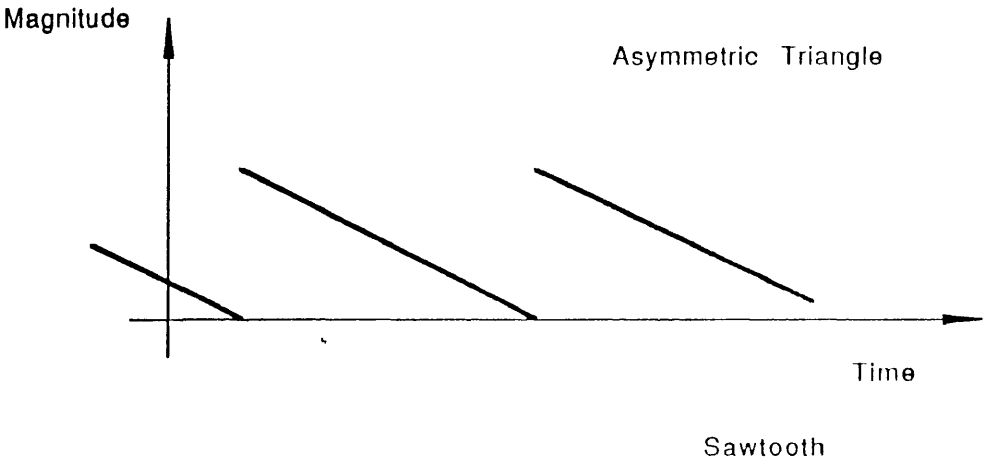
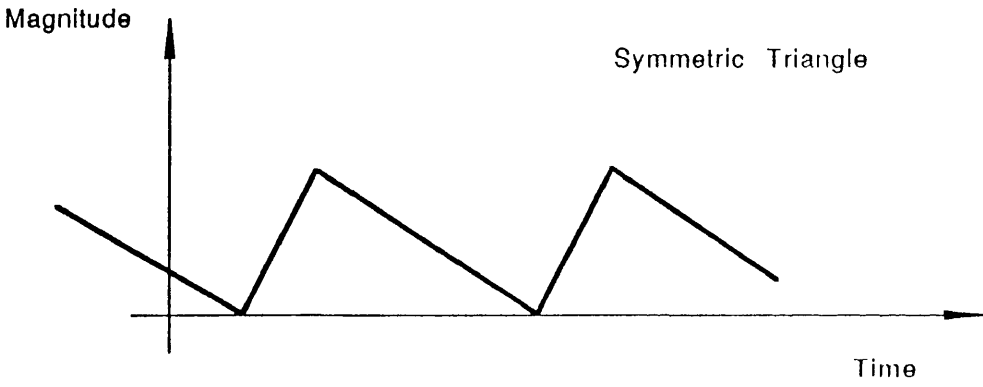
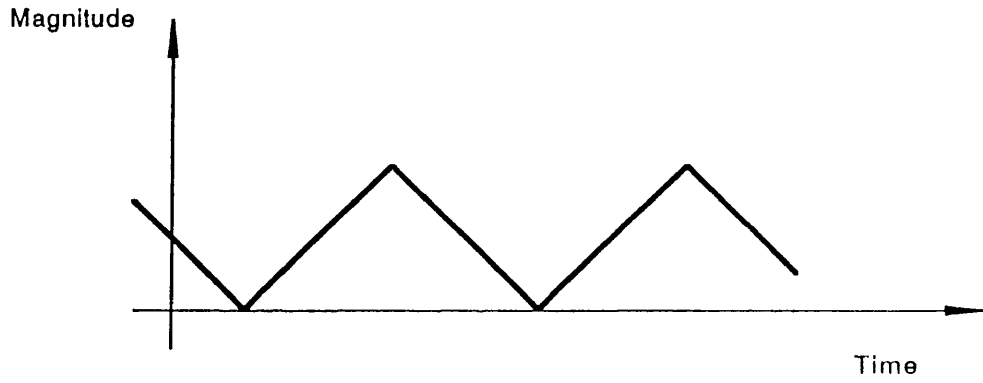


Fig. 8: Ramp Waveforms

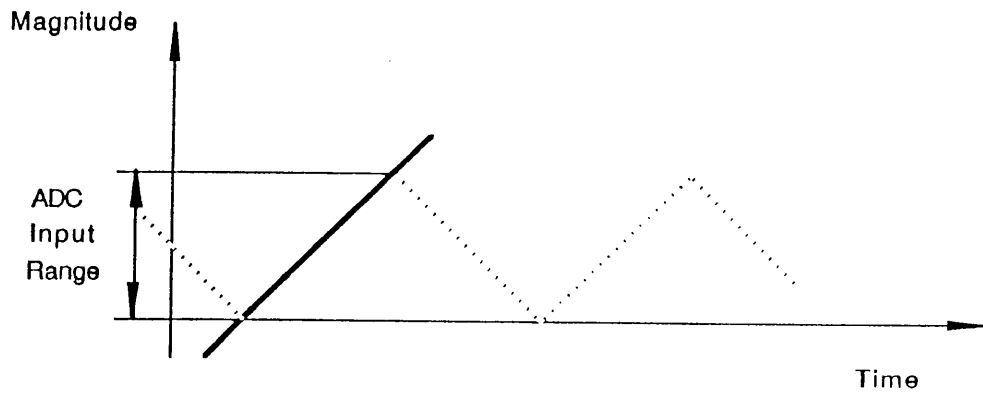


Fig. 9: Single Sweep

The value of k can easily be determined experimentally, without prior knowledge of either f_s or b . It is sufficient to count the total number of samples (during the sweep) that yield output codes between 1 and $2^N - 2$. Dividing this number, M' , by the corresponding number of lsb , $2^N - 2$, yields k .

$$k \approx \frac{M'}{2^N - 2} \quad (35)$$

It is clear that if a histogram is made during the sweep, the expected values of the bin widths W_C ($C = 1 \cdots 2^N - 2$) of the ADC will be proportional to the tallies M_C .

$$W_C = \frac{M_C}{k} \quad (36)$$

However, since the input signal is deterministic rather than random, the counting process does not have the characteristics of a dichotomy anymore, and equation (18) is not valid any longer to calculate the standard deviation. Instead, the following reasoning should be used. Since the input signal is such that there are k samples per lsb , the expected number of samples with input signal values within bin C , of width W_C , will be $k W_C$. Unfortunately, k is not always an integer value. The actual number of samples within bin C will be M_C . It is clear that the following relation will hold:

$$\text{int}(k W_C) \leq M_C < \text{int}(k W_C) + 1 \quad (37)$$

The maximum, absolute error on M_C is ± 1 . As a result, the maximum absolute error on W_C (given by equation (36)), is

$$\epsilon_{W,max} = \pm \frac{1}{k} \quad (38)$$

If k (the expected number of samples per lsb , or per bin) is large enough, one could assume that this error is spread evenly, and the standard deviation σ on W_C could be expressed as

$$\sigma^2 = \int_{-1/k}^{1/k} p(\epsilon) \epsilon^2 d\epsilon = \int_{-1/k}^{1/k} \frac{k}{2} \epsilon^2 d\epsilon = \frac{1}{3 k^2} \quad , \quad \text{or} \quad \sigma = \frac{1}{k} \frac{\sqrt{3}}{3} \quad (39)$$

Obviously, this will also be the standard deviation on the DNL values, derived from W_C by subtracting 1. Since $k = M'/(2^N - 2)$, the expression can be expanded to

$$\sigma = \frac{2^N - 2}{M'} \frac{\sqrt{3}}{3} \quad (40)$$

The minimum number of samples required to obtain all DNL values with a standard deviation of σ , is given by

$$M'_{min} = \frac{2^N - 2}{\sigma} \frac{\sqrt{3}}{3} \quad (41)$$

This is normally much less than the number of samples required if the histogram test had been performed using a truly random input signal (equation (25)), especially for small values of σ . The factor σ now appears to the power of -1 rather than to the power of -2 .

The difference is even more significant where INL is concerned. One can easily verify that since inl_i is the error on the segment of the input range for which the output code C is between 1 and i (or the union of bins 1 through i), the error can be determined in a way similar to the determination of the error on one bin. As a result, equations (40) and (41) also apply to the standard deviation on the experimental determination of the INL.

Although the situation appears more favorable than in the truly random case (equation (25)), it needs to be stressed that the expression for σ , derived for the single sweep, only apply when the sweep speed is chosen such that a large enough number of samples falls into each bin. In practice, k must be at least 5 to 10. Since $k = f_s/b$, the minimum duration of the sweep must be at least $(2^N - 2)/b = k (2^N - 2)/f_s = k (2^N - 2) T$, with T one sampling period. If the sweep is considered as one half period of a periodic ramp signal, the ramp frequency should be equal to $f_s/(k (2^{N+1} - 4))$. Since this is a very low frequency compared to the Nyquist frequency $f_s/2$, the single-sweep method is only appropriate for measurements of the static (DC) transfer curve.

Another important point about the single-sweep method is that the sweep must really be taken at once. It cannot be interrupted, or reconstructed from several different sweeps. As a result, the measurement system must have the

capability to store at least $5 \cdot 2^N$ output codes in a real-time fashion, which is practical for converters with resolutions up to 12-14 bits.

One could wonder why we derived an expression for σ (equation (40)), while we already had a formula for the *maximum* error (equation (38)). The answer is that very often, a histogram test is performed using not a single sweep, but several periods of a triangular waveform. In that case, the different sweeps (half periods) can usually be considered statistically independent, and equation (40) can be used to calculate the standard deviation on the total histogram. The overall standard deviation will be equal to the standard deviation of one equivalent sweep (predicted by equation (40)), divided by the square root of the number of sweeps. Although this situation is still more favorable than taking an equal number of samples of a truly random input signal, the expected error quickly reaches the same value as was derived for the random case when the number of sweeps is increased, especially if the number of samples from each sweep falls below $2^N - 2$ (number of bins).

In practice, misconceptions exist as to how many samples are needed in order to determine static INL values from a histogram test. Many people use a rule of thumb similar to equation (31), although they apply a low-frequency deterministic input signal to the ADC. This obviously represents a waste of measurement resources. Actually, the most efficient way to perform a histogram test, is to use one low-frequency sweep containing many samples per *lsb*.

2.4. Distortion of the Ramp Waveform

Although linear ramp waveforms (or a single linear sweep) offer definite advantages when used to determine the linearity of an ADC through histogram analysis, their use has some drawbacks. In particular, the results of the histogram test performed as described above, will only reflect the ADC transfer curve in an unbiased way if the input signal is *truly* linear. In practice, this may be a limitation, since commonly available signal generators usually only produce ramps of limited accuracy (the equivalent of 9-12 bits linearity) at low frequencies. Especially the tops (break-points) of the ramp are subject to distortion, but the nominally linear portion may be affected as well. The linearity often degrades even further at higher frequencies.

Even when the signal generator has sufficient linearity for the measurement to be performed, the ramp can easily become corrupted by the measurement set-up. In particular, limited frequency response or reflections in cables, printed circuit boards, IC pins etc., can reduce the linearity of a ramp before it even reaches the input of the ADC. This is usually only of concern when one tries to evaluate an ADC for relatively high frequency input signals (dynamic behavior), rather than trying to measure the DC transfer curve. However, it will be shown in a later section that histogram tests performed under dynamic conditions may reveal very useful information.

The influence of frequency-limiting factors in the set-up can easily be estimated when the effective bandwidth of the set-up is known. As an example, we could represent the cabling as a first-order low-pass filter, with time constant τ . Using elementary linear analysis techniques, a time domain filter response as depicted in figure 10 can be found. The linear portion of the output signal will be delayed slightly with respect to the input ramp, which is of no concern for the histogram test. But in addition, the top of the waveform will be "flattened" in an exponential way. The decaying exponential has a time constant of τ , and can be written as

$$f(t) = A e^{-\frac{t}{\tau}} \quad (42)$$

The initial slope, $\frac{dV}{dt}_{init}$, is $-A/\tau$, but is also equal to the negative slope of the ramp, minus the positive slope, which for a symmetric ramp of frequency f , yields

$$\frac{dV}{dt}_{init} = -4Rf \quad (43)$$

The initial value A of the decaying exponential can be found by equating expressions for the initial slope

$$A = -\tau \frac{dV}{dt}_{init} = 4\tau Rf \quad (44)$$

The amount of time t_N needed in order for the transient to decay below the N bit level, can be found from the expression

$$A e^{-\frac{t}{\tau}} = 4\tau Rf e^{-\frac{t}{\tau}} < \frac{R}{2^N} \quad (45)$$

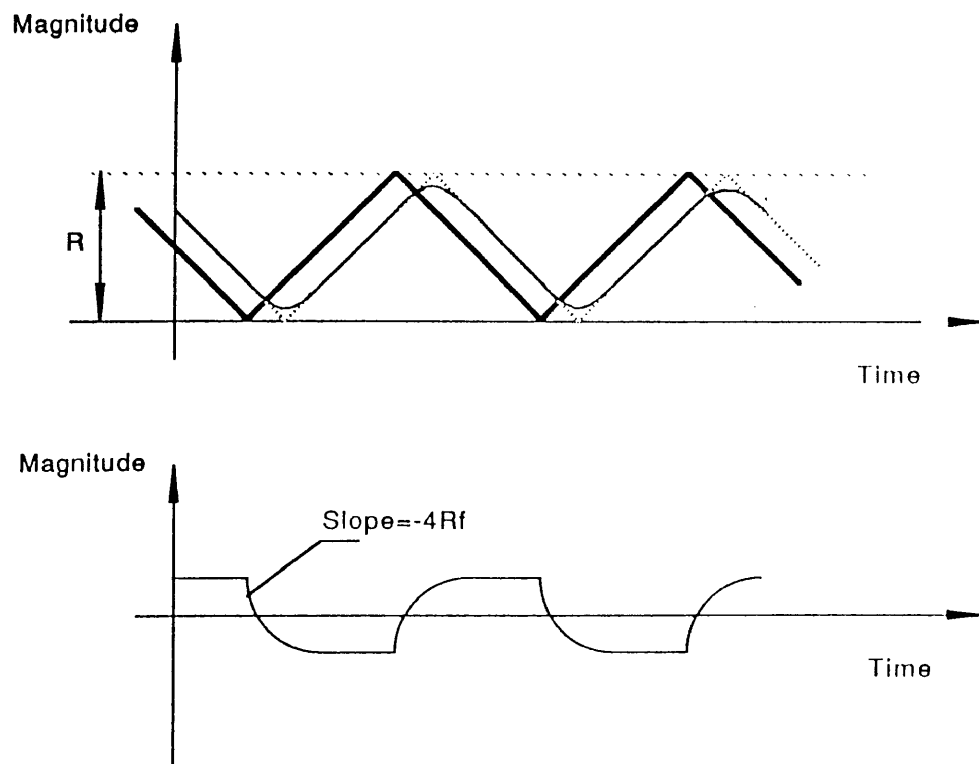


Fig. 10: Ramp Response

Or:

$$t_N > \tau \ln(\tau f 2^{N+2}) \quad (46)$$

The fraction x of the ramp that should be discarded due to flattening of (each) top, is

$$x = \frac{t_N}{T/2} = 2f t_N \quad (47)$$

If the necessary precautions are not taken to discard the tops of the ramp, the flattening will cause an increase in the probability density of the signal at high or low magnitudes, and the histogram will be biased towards a "U" shape, as shown in figure 11. Obviously, DNL and INL values would be biased as well.

It should be mentioned that the frequency response of cabling used in a measurement set-up does not usually fit the first-order filter model. However, reasonable estimates of the "bad" portion of the ramp can still be obtained using the formulas above, if a time constant is used corresponding to the effective bandwidth of the overall signal path, from signal source to ADC input stage.

2.5. Histogram Analysis Using Sine Waves

The problems associated with flattening of the tops of a ramp can be avoided if sine waves are used instead. It is well-known that the response of a linear system to a sine wave is always a sine wave, no matter how many parasitic resistive or reactive components may be present in the system. Another advantage is that high-frequency sine waves can be generated with a great amount of precision, due to the fact that highly selective linear filters can be built into the function generator.

The procedure for the histogram test, described above, can be modified in order to accommodate the use of sine waves instead of linear ramps. The major difference is that a sine wave does not have a constant magnitude density function, like a ramp does. A sinusoidal signal can still be used in order to perform a histogram, but the histogram will have to be compensated for this fact before DNL and INL are calculated.

We will assume that a histogram is made using an ADC, with the input signal (in this case a voltage) given by

$$v_{in} = v_{off} + A \sin(\omega t + \phi) \quad (48)$$

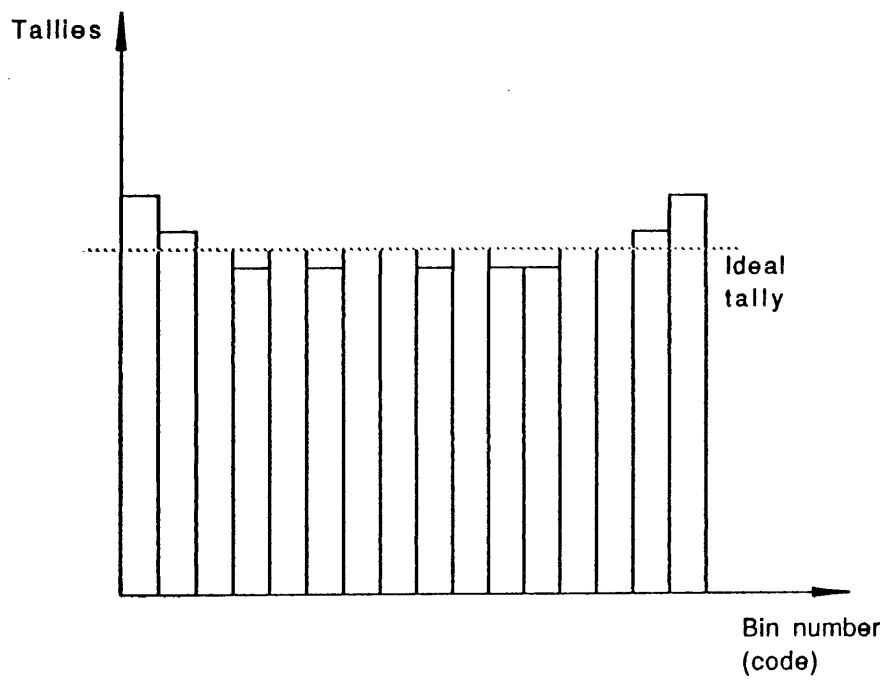


Fig. 11: Histogram with U-Shaped Bias

With v_{off} the offset voltage, A the magnitude, t the time and ϕ the initial phase angle.

It can be shown that if this signal is sampled at random times, the probability density of the voltage for each sample would be

$$p(v) = \frac{1}{\pi A \sqrt{1 - \left(\frac{v - v_{off}}{A}\right)^2}} \quad (49)$$

The probability that the input voltage would be between v_1 and v_2 , is

$$P(v_1, v_2) = \int_{v_1}^{v_2} p(v) dv \quad (50)$$

$$P(v_1, v_2) = \int_{v_1}^{v_2} \frac{1}{\pi A \sqrt{1 - \left(\frac{v - v_{off}}{A}\right)^2}} dv = \frac{1}{\pi A} \left(\text{asin}\left(\frac{v_2 - v_{off}}{A}\right) - \text{asin}\left(\frac{v_1 - v_{off}}{A}\right) \right) \quad (51)$$

The probability that an input voltage would be in bin i of the ADC, is

$$P(\text{bin}_i) = \frac{1}{\pi A} \left(\text{asin}\left(\frac{ctl_{i+1} - v_{off}}{A}\right) - \text{asin}\left(\frac{ctl_i - v_{off}}{A}\right) \right) \quad (52)$$

This formula could be used to compensate the histogram tallies before the DNL values are calculated. Each tally should be divided by (52) and multiplied by $1/(2^N)$. The constant $\pi A/2^N$ can be dropped, if M' is later replaced by the sum of normalized tallies, since the probabilities are calculated as *ratios* of numbers of samples within a bin to total number of samples. However, the difficulty is that v_{off} and A first have to be estimated, while the *actual* code transition levels are unknown (the purpose of the histogram is precisely to calculate them).

The estimation of v_{off} and A can be performed by a non-linear (4-point) fit [6], applied to the data record in the digital domain. The values will obviously be found in lsb. The reasoning is that although the ADC is not necessarily perfect, a sine wave will still be digitized as a sine wave, only *slightly* corrupted by quantization noise and non-linearity. The 4-point (iterative) fit method has excellent immunity against these effects.

More problematic is the estimate of ctl_i and ctl_{i+1} . Since the purpose is to normalize the histogram for the probability density of the sine wave versus constant probability density and *not* for the fact that the ADC could be inaccurate, the values of ctl_i and ctl_{i+1} should be spaced 1 *lsb* apart. In most cases (ADC with reasonable number of effective bits), a negligible error is made if the ideal values ($ctl_i = i \text{ lsb}$, $ctl_{i+1} = (i + 1) \text{ lsb}$) are chosen. In critical situations, a first estimate of DNL and INL could be obtained this way. From this information, actual code transition levels (in *lsb*) can be derived. These new estimates could then be used to normalize the histogram again, e.g. using the newly found value for ctl_i , and the same value plus one *lsb* for ctl_{i+1} . Each tally should now be divided by (52), with the new values for each set of ctl_i . One or more new estimates could be done in an iterative fashion, in order to improve accuracy, but this is rarely needed.

One could derive formulas for the standard deviation on DNL and INL

values found through a histogram test using sine waves. These formulas would be fairly complicated, since the theoretical probability that a sample would fall in one bin, is not constant. However, a simple worst-case approach can be taken. One can calculate the expected number of samples in the bin that corresponds to the offset value of the sine wave (the center of gravity of the magnitude distribution). Since this bin has the smallest number of expected samples, application of the formulas for a ramp waveform, using this expected number of samples in that bin, will yield a slightly pessimistic, but safe measure of accuracy.

2.6. Noise in a Histogram Test

Without going into mathematical details, it can easily be shown that noise will not influence a histogram test to a great extent. Noise can be introduced into the measurement from several sources. The signal generator may be noisy, the cables or test set-up may generate noise and finally, noise may originate within the ADC itself, due to thermal agitation of charge carriers within semiconductor devices and resistors. The effect of these different kinds of noise will be essentially similar: the noise will be added onto the analog signal that is converted. Although different in nature, interference from external signal sources (60 Hz, heavy machinery, high-frequency switching of computers...) can also be considered as noise, as long as it is not correlated with the input signal.

The effect of noise (especially high-frequency noise) will be moderated during the histogram test, since the noise signals normally have zero mean. As a result, noise may occasionally force a sample from one bin to another, but chances are that at some later time, another sample may be forced back to that same bin from another one. Since the process is zero-mean, noise will not introduce a significant bias on the histogram values, especially not when a constant probability density signal is used (e.g. linear ramp).

However, one should keep in mind that the effect of noise could be to increase the uncertainty (standard deviation) on estimated DNL and INL values, beyond what could be expected from the formulas derived earlier. If a truly random, constant pdf signal were applied to the ADC for the purpose of a histogram test, noise would not affect the estimation of DNL or INL values, and formulas (24) and (29) would still be exact. The reason is the following. The addition of a zero-mean random signal to another random signal creates a random signal with a pdf equal to the convolution of the respective pdf's, which within the nominal input range would still be constant (figure 12).

However, if a single-sweep method is used, noise will alter the standard deviation on DNL and INL. The swept signal is a deterministic signal, while the noise is random. Depending on the magnitude of the noise signal, this will make the input signal more random, and equation (38) will not correctly reflect the estimation error. As a matter of fact, the larger the noise signal, the more the single-sweep situation would resemble the random signal situation.

Exact mathematical formulas to quantify the uncertainty of DNL and INL are hard to obtain, especially if the magnitude of the noise exceeds one *lsb*. However, an experimental criterion can always be applied. In order to determine the standard deviation on the tallies of a histogram using the single-sweep method, one sweep could be immediately followed by a second one. The respective histograms could be compared, bin by bin, and the root mean square difference (divided by 2), used as an estimate for the standard deviation on the histogram tallies M_C . This standard deviation could be included into the formulas for DNL and INL, to reflect the combined influence of a noisy input signal, *and* random sampling.

If it is technically difficult to perform two consecutive sweeps (the sweeps should not be separated by a long time, since in precision, self-calibrated convert-

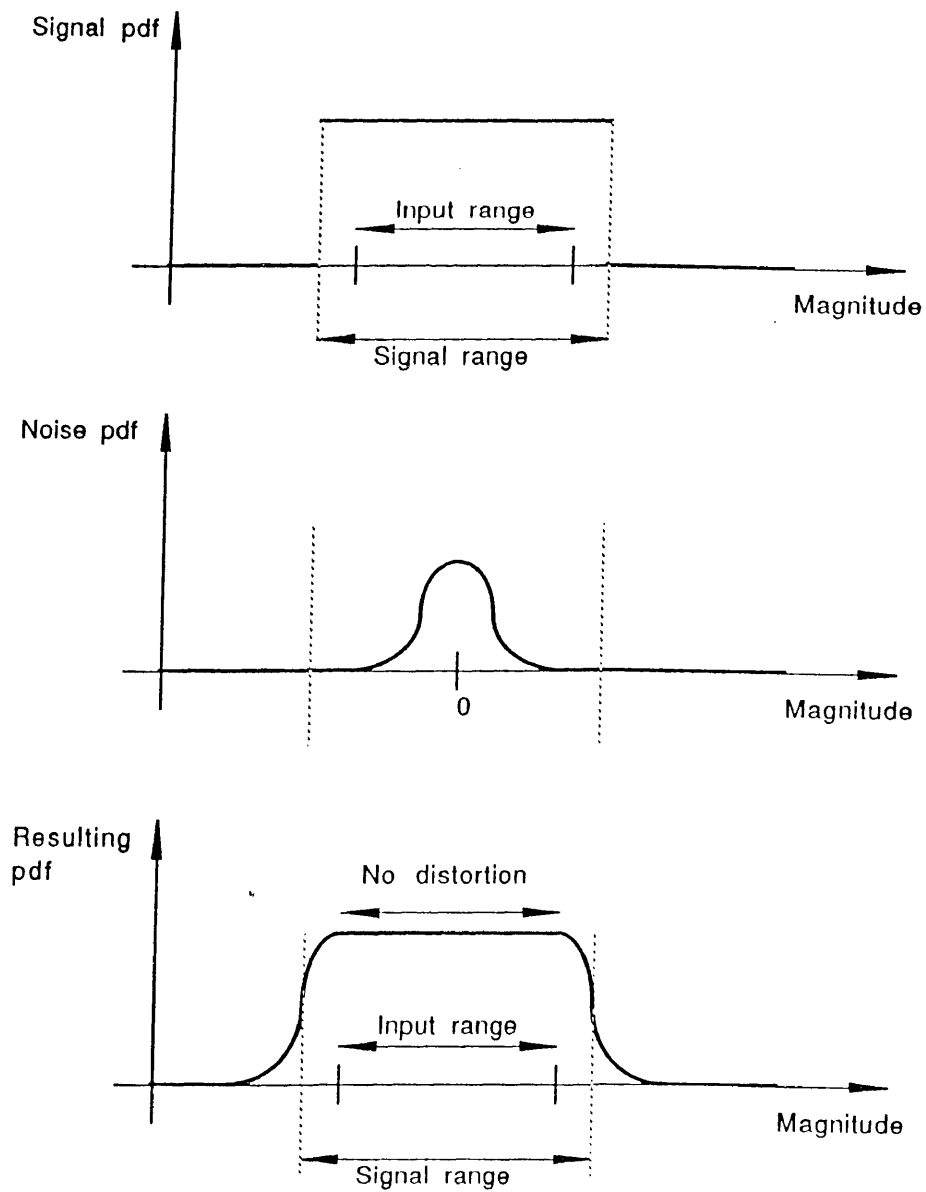


Fig. 12: Probability Density Function Affected by Noise

ers, component values may have drifted), two histograms with half the samples could be constructed by assigning consecutive samples first to one, and then to the other histogram, in an alternating fashion.

Another method of estimating the influence of noise, is to connect the input of the ADC to a fixed, DC signal source (e.g. analog ground). A histogram can be constructed with the output data. This histogram will reflect the magnitude distribution of the noise (in *lsb*), and this information can be used to calculate the uncertainty on an actual histogram test. However, the calculation will involve the convolution of two probability densities (the signal and the noise) and will not be as straight-forward as the previous methods.

2.7. Combining Histograms

So far, it has been assumed that a histogram test was always performed over the complete range of the ADC. This is probably the most practical procedure, as long as high-quality input signals are available, that can cover the complete input range without noticeable signs of distortion. However, this may be a limitation, especially when the histogram is determined using a linear ramp.

One solution to this problem, is to assemble the histogram from a number of partial histograms, measured over a limited input signal range. However, some caution needs to be exercised when combining different measurements.

The traditional histogram test consists of tallying the number of occurrences of each output code, between 0 and $2^N - 1$. The tallies for these two extremes are ignored. If the input signal is a ramp, the next step is to calculate the normalized bin widths, by dividing each tally by M' (number of samples between 1 and $2^N - 2$), and multiplying by $2^N - 2$. DNL and INL then follow.

A slightly modified procedure can be followed to derive the same information from two overlapping histogram tests. We will assume that the first test tallied the output codes 0 through Y , while the second, independent test tallied the codes X through $2^N - 1$, with X and Y integers such that $0 < X < Y < 2^N - 1$. This is shown in figure 13.

Before the two histograms are combined, they should be normalized so as to have the same density of samples in the portion between X and Y . This can easily be achieved by dividing all the tallies in the first histogram by M'_1 .

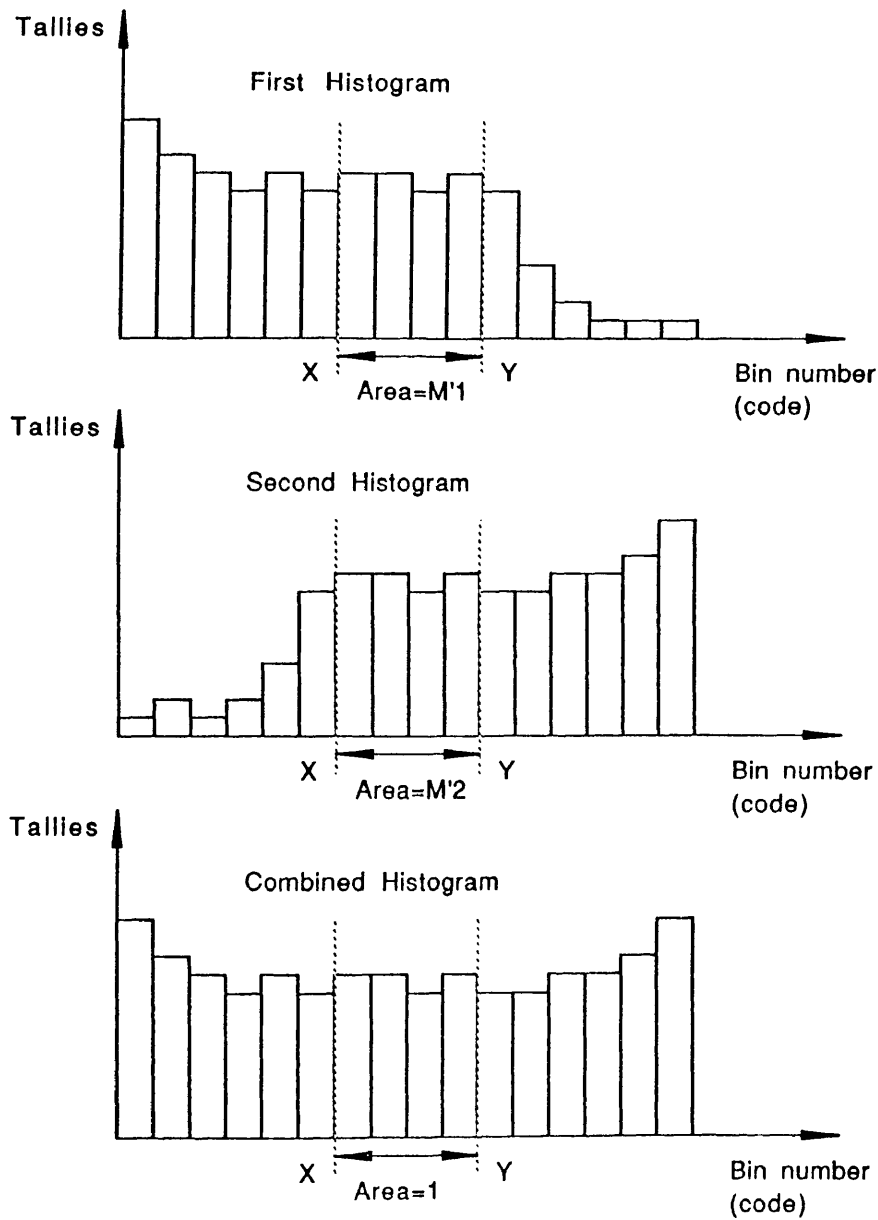


Fig. 13: Combining Two Histograms

and all the tallies in the second histogram by M'_2 . M'_1 is the number of tallies with codes between X and Y in the first histogram. M'_2 is the number of tallies with codes between X and Y in the second histogram. After this operation, the two histograms become compatible, and the standard procedure can be applied to calculate bin widths, DNL and INL. The same principle could be extended towards the use of different sine waves, but the complexity of such operation may be excessive.

2.8. Harmonically Related Input Signals

Several approaches exist concerning the choice of input signal and ADC sampling frequency for a histogram test. The general theory was

derived based upon a *random* input signal with known (e.g. uniform) magnitude distribution. Unfortunately, truly random signals of this nature are hard to generate. It has also been shown that for a truly random signal, the estimation error on the different probabilities was not very favorable. A better approach is to perform the histogram test using a single sweep of the input signal, while the ADC sampling rate is held constant.

The disadvantage of this method is that the frequency of the input signal is always at least 2^{-N} times lower than the sampling frequency. In some cases, it may be desirable to increase the input

signal frequency (e.g. for dynamic testing). If the ADC sampling rate is held constant, the process will obviously not be truly random either. This situation could be modified by taking samples at random times, but again, doing so may be impractical or undesirable.

In practice, a histogram test is almost always performed using a deterministic input signal of fixed frequency f , *and* a fixed (deterministic) sampling rate f_s . The frequency ratio f_s/f has serious implications concerning the relevance of the information obtained from the histogram test. When this frequency ratio is equal to an exact ratio of integers, the input signal is said to be harmonically related to the sampling frequency.

If the ratio is an exact integer number (e.g. 10), one can easily see that the same points of the input signal will be sampled over and over again (figure 14). The histogram of such measurement will appear as 10 huge peaks (figure 15).

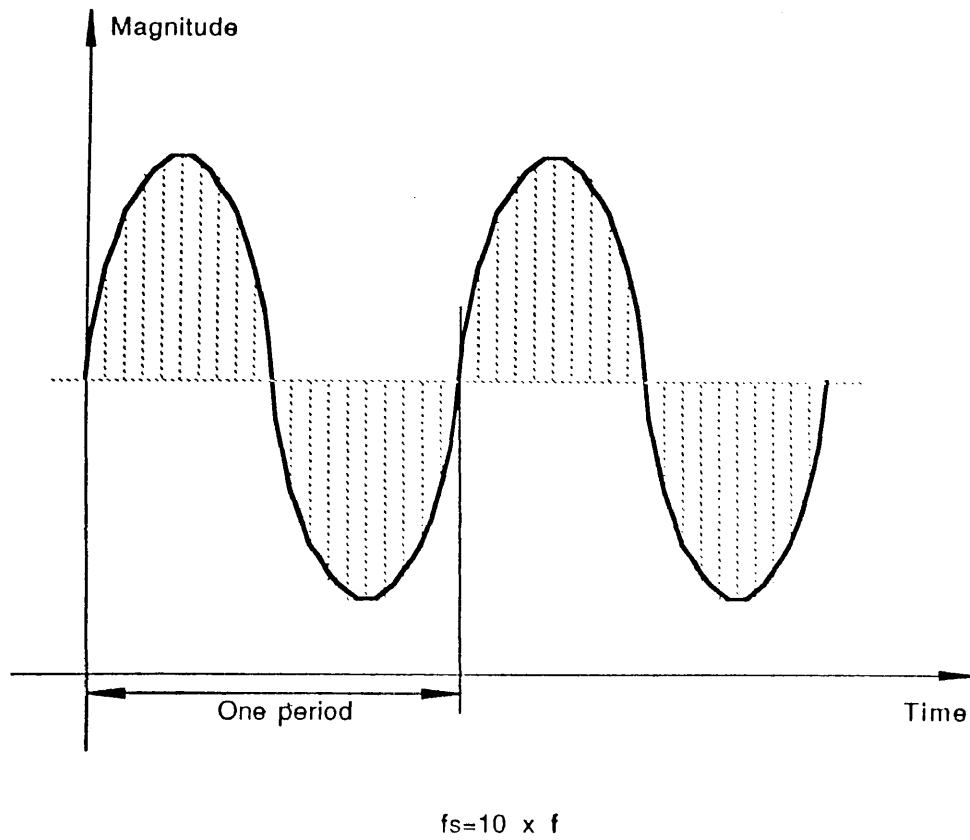


Fig. 14: Harmonically Related Input Signal

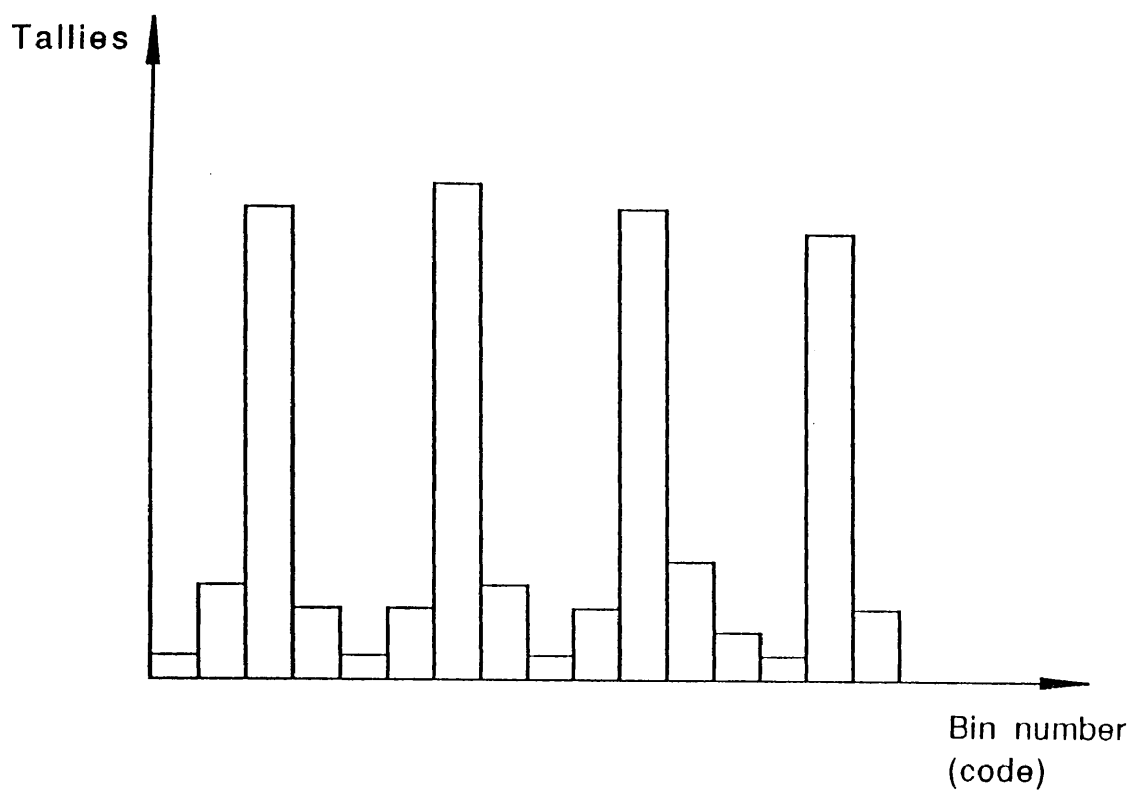


Fig. 15: Histogram of Harmonically Related Waveform

rather than a smooth curve, and the results will be meaningless.

Something similar happens if the frequency ratio is equal to the ratio of two prime numbers (e.g. $17/3$). In that case, 17 different samples will be taken out of 3 periods of the input signal, and the process will again repeat itself, without adding new information. In other words, if 100,000 samples are taken at a frequency ratio of exactly $17/3$, no more information will be available than if only 17 samples had been taken.

As a general rule, the maximum number of *effective* (significant) samples corresponding to a frequency ratio of r can be found by reducing r to a fraction of two integers, n/m , with m the smallest possible integer denominator. The effective number of samples is then n . This value should be used to calculate the estimation error on the histogram (using the formulas for deterministic input signals), rather than the actual total number of samples taken. If the input waveform has half-period symmetry (like is the case for a sine wave or a symmetric ramp, the situation could even be worse, with the effective number of samples equal to $n/2$).

One can verify that the frequency ratio for a linear, single-sweep input signal with L samples taken, would be $2L/2 = L/1$. In other words, maximum benefit is obtained from the number of samples taken.

Another effect associated with the frequency ratio, is systematic bias of the histogram due to unfinished periods. An example is shown in figure 16. If the frequency ratio is 10, and 22 samples are taken, two of the 10 different codes will occur three times, while the other eight only occur twice. This will bias the histogram. The effect can be avoided by truncating the record of samples taken (nominal length L) to exactly $n \text{ int}[L/n]$ samples. Again, one can verify that the single-sweep method automatically satisfies this condition.

The frequency ratio r can be calculated from external measurements with a very precise frequency counter, or from non-linear fitting of a sine wave (assuming sine wave input), performed on the output data record. Methods using the fast Fourier transform in the digital (output) domain are possible as well.

2.9. Fourier Analysis

The purpose of the histogram test is to eventually calculate normalized bin widths, DNL or INL. These are three different, though equivalent representations

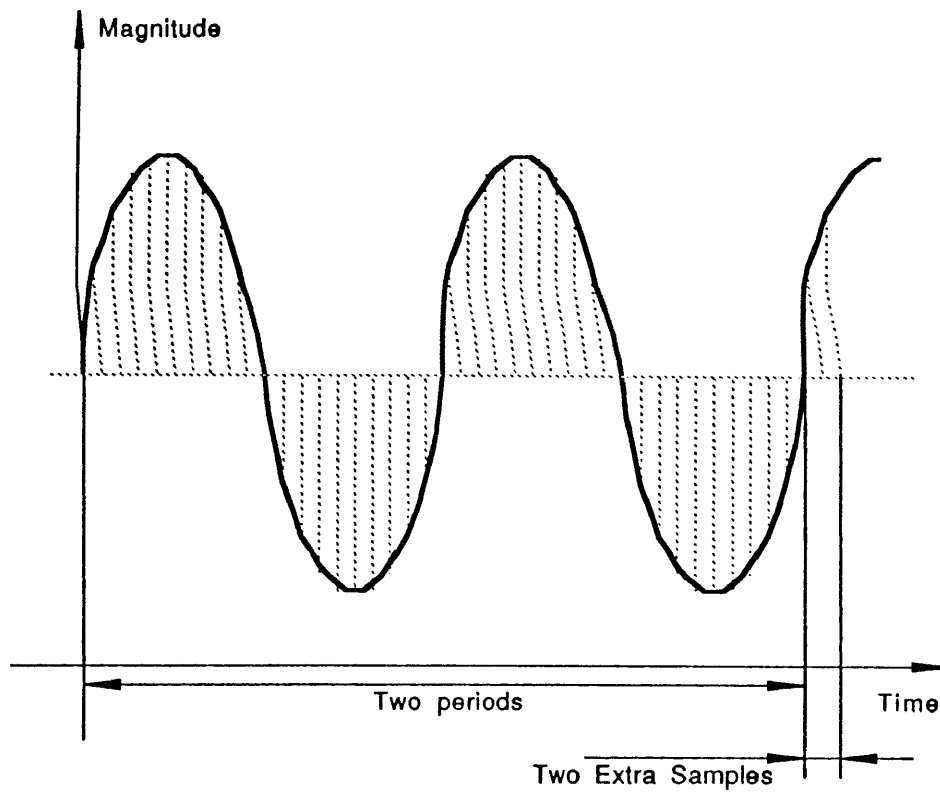


Fig. 16: Unfinished Period

non-idealities in the ADC the transfer curve. The effect of such non-linear transfer curve is that when an arbitrary input signal is applied to the ADC, its digital representation will suffer a certain, systematic distortion.

A common way to evaluate such distortion directly (without first measuring the transfer curve), is to apply sinusoidal signals to the ADC, and to determine the distortion from the digital representation of the sine wave. This can be performed fairly easily on a digital computer, by calculating the fast Fourier transform (FFT) of an output record of a certain length. In order to eliminate the effect of finite record length, the record should first be "windowed", which is a standard operation [9,4,6]. Eventually, the FFT output will show a peaked spectrum (figure 17). The first peak represents the input signal (fundamental) frequency. Other peaks, at multiples of this fundamental frequency, represent the distortion. By adding the energy of all the distortion peaks and dividing this by the energy of the fundamental, the signal to distortion ratio (SDR) can be calculated.

The FFT output also shows a noise floor. This is due to two different effects. First, there is the inherent quantization error of the converter. Although this error is strictly speaking a deterministic function of the input, in practice (especially for high-resolution converters), its value can safely be considered a random variable, with values uniformly distributed between 0 and 1 lsb (ideal converter). The expected energy of the quantization error can be calculated as the variance (σ^2) of this distribution, which is

$$\sigma^2 = \int_0^1 1 (x - 0.5)^2 dx = \frac{1}{12} \quad (53)$$

This energy is distributed over the L bins of the FFT, yielding a theoretical quantization noise floor of $1/(12 L)$. In practice, noise inherent to the ADC circuitry (semiconductor and resistor noise) is added to this value, causing the noise floor to rise. The total noise energy can be determined from the FFT output, by multiplying the average noise floor energy (per bin) by the number of FFT bins. The ratio of this number to the energy of the fundamental peak, expresses the signal to noise ratio (SNR) of the converter, for that particular input sine wave. When noise and distortion energy are added together and divided by the energy of the fundamental, the signal to noise and distortion ratio (SNDR) is obtained.

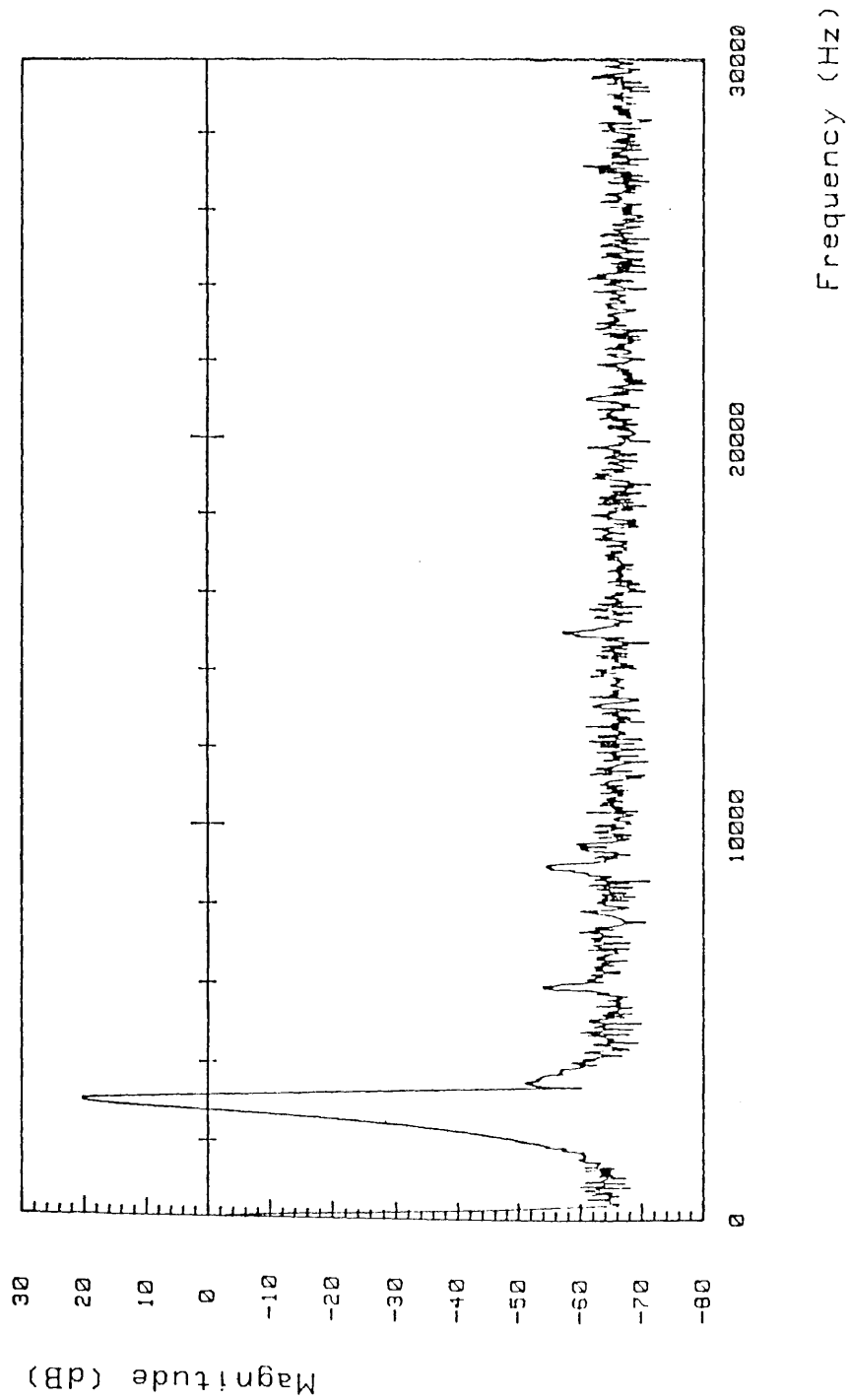


Fig. 17: Typical FFT Response

The advantage of FFT characterization is that many effects (distortion due to a non-linear transfer curve, noise etc.) are combined into one, relatively easy to perform measurement. The record length can be fairly short, in practice 1024, 2048 or 4096 points. However, one should keep in mind (although this is often overlooked in practice) that earlier considerations concerning frequency ratio f_s/f *also* apply to an FFT measurement. If a 10 bit ADC is characterized using the FFT method on a record of 1024 samples, and the frequency ratio is 50, only 50 *different* samples will be obtained, and the FFT (or SDR or SNR) will *not* reflect *all* non-idealities in the transfer curve.

Another advantage of the FFT method, is that dynamic effects within the ADC circuitry are included in the measurement when a high-frequency sine wave is applied. Dynamic effects are, in general, undesirable, since they limit the maximum input signal frequency for which the ADC will exhibit linear behavior. The origin of some of these dynamic effects will become apparent during the discussion of actual ADC architectures. In order to design very fast ADC's with a high input signal frequency range, it is important to be able to evaluate the limitations within existing ADC's. The FFT method, applied at different input signal frequencies, makes it possible to detect the presence of dynamic distortion.

However, a major limitation of the FFT method is that, since many effects are combined, the precise cause of imperfect behavior cannot be traced. As a result, the FFT method provides an excellent testing tool (provided some precautions are taken regarding number of samples and frequency ratio), but a very limited diagnostic tool for ADC evaluation.

2.10. Conclusion

This chapter introduced two important ADC testing methods: histogram analysis and fast Fourier transform (FFT). The histogram procedure was derived from the concept of bin width and differential non-linearity. The theory was first developed for random input signals, using statistical concepts. The same concepts were used to derive practical formulas relating the accuracy of non-linearity values (DNL and INL) to the number of samples taken. Similar formulas were derived for situations where the input signal is not truly random, but a deterministic waveform, which is normally the case in practical measurements. Both the use

of linear ramps and sine waves were discussed, as well as the advantages and disadvantages of both methods. One section was devoted to the influence of signal frequency and sampling rate on the accuracy of the histogram test. Finally, a discussion of the FFT method was given, including its most obvious advantages and disadvantages.

CHAPTER III

DYNAMIC HISTOGRAM TESTING

3.1. Introduction

In this chapter, some shortcomings of the traditional histogram test method are discussed. In particular, it is demonstrated that the traditional method can only accurately model the *static* ADC transfer curve. When high-frequency input signals are used in order to perform a histogram test, the result may be different than what is observed at low frequencies. These are called dynamic effects, whose existence is well-known, but whose effects are only now being investigated in more detail [10,11].

Due to the high input signal frequency, an additional degree of freedom becomes significant within the system, being the time derivative (slope) of the input signal. The slope can excite dynamic error mechanisms within the converter, which affect the measurement results. Although it is known that histogram measurements are affected by high-frequency input signals [4, 6], the actual way in which this happens has not been thoroughly investigated before.

However, if the histogram concept is extended, the influence of the first derivative can be included in the picture, and valuable dynamic test results can be obtained. It will be shown that any periodic input signal can be modeled as a closed locus in a two-dimensional magnitude-slope plane. By modeling the histogram domain in a similar (two-dimensional) way, non-linearity measurements can be obtained which include dynamic effects due to the slope of the input signal. This novel procedure provides much more accurate modeling than either the traditional histogram or the FFT method. If the effect of higher derivatives is neglected, this model will accurately reflect the full, dynamic performance of a specific ADC.

3.2. Dynamic Errors

It has been shown that the FFT test provides a convenient way to characterize dynamic ADC behavior in a qualitative way. However, the test does not reveal the actual transfer curve of the ADC under dynamic conditions. The histogram test

lends itself extremely well to the measurement of the static ADC transfer curve. However, it requires that the input signal be of sufficiently low frequency, so that dynamic effects within the ADC would not corrupt the measurement.

It will be shown later, that static DNL and INL curves of an ADC, as determined through a low-frequency histogram test, provide an excellent diagnostic tool to find out what DC errors cause imperfect behavior within multi-stage ADC's. One could wonder if similar results could be obtained out of a high-frequency histogram test. The answer is yes, but not in the classic sense. In order to perform dynamic characterization, the histogram concept needs to be generalized, due to additional degrees of freedom of the input signal over frequency.

Histogram tests have commonly been performed with high-frequency sine waves, using a normalization procedure as described earlier. It has been observed that from a certain frequency on, the shape of the histogram started changing, due to the influence of dynamic errors (and hence non-linearity) within the ADC. This method has been used to determine the maximum signal frequency (in the sine wave sense) that could be applied to an ADC without introducing significant distortion. Similar distortion of the histogram could be observed when using high frequency ramp signals, but to our knowledge, no attempt has ever been made to correlate the two phenomena.

As a result of insufficient understanding, a lot of confusion exists as to how a histogram test is affected by the use of sinusoidal signals or (relatively) high-frequency ramps. Many people have almost given up on performing such histogram tests, and instead prefer to use the FFT method for dynamic evaluation of ADC performance. This is almost unfortunate, since a high-frequency histogram test can reveal fairly complete information about dynamic errors, which FFT characterization cannot.

The difference between a low-frequency (ideally DC) sweep and a high-frequency ramp input signal, is their time derivative. In both cases, the signal will assume all possible input signal values (at least if the magnitude of the sweep or ramp is sufficient to cover the input range). However, in the first case, the time derivative (slew) is practically zero, while in the second case, it is $\pm 2Af$, with A the peak-to-peak magnitude of the ramp and f the frequency. It is clear that if the frequency of the sweep is increased, the time derivative will increase accordingly.

Similarly, the difference between a low-frequency and a high-frequency sine wave, as far as the ADC is concerned, is a difference in the time derivative(s) associated with each signal level.

Since the internal circuitry of the ADC is not perfect, the actual code transition levels may not be equal to the ideal code transition levels. The amount of variation in the actual transition levels is normally determined by several factors. One of them is the particular value of the input signal (in the DC or low-frequency sense). This is what is measured when a DC characterization is performed, since INL is expressed for each bin, and the bin number reflects the magnitude of the input signal. However, in the general sense, the code transition levels (and hence INL values) are more complicated functions of the input signal, involving not only the magnitude, but also a number of higher time derivatives.

The presence of the derivatives in the generalized INL function reflects the dynamic behavior of the ADC. It can be due to limited frequency response of amplifiers, limited switching speed of comparators, comparator hysteresis, or signal-dependent timing uncertainty in sampling devices within the system. In most cases, the dependence is mainly on the first derivative of the signal, and the influence of higher derivatives can safely be neglected. However, the influence of the first derivative has implications upon the way a high-frequency histogram test should be performed.

During a traditional histogram test, it is implicitly assumed that all derivatives of the signal are always zero. If a high-frequency input signal is applied to the ADC, this assumption will not hold anymore, and INL values will not be a function of the signal magnitude alone. As a result, different histograms, and hence different INL values will be found when the signal frequency is changed. In order to resolve this ambiguity, new independent variables should be introduced, in practice at least the first derivative of the signal.

In order to visualize this concept, the input signal could be represented as a locus of values in a two-dimensional signal plane. The first dimension is the signal magnitude, x . The second dimension is the time derivative of the signal, x' . Any momentary value of the input signal and its derivative can be represented as a point in the plane. The locus of the signal is the continuous curve, obtained by connecting the different values the signal assumes over time (figure 18).

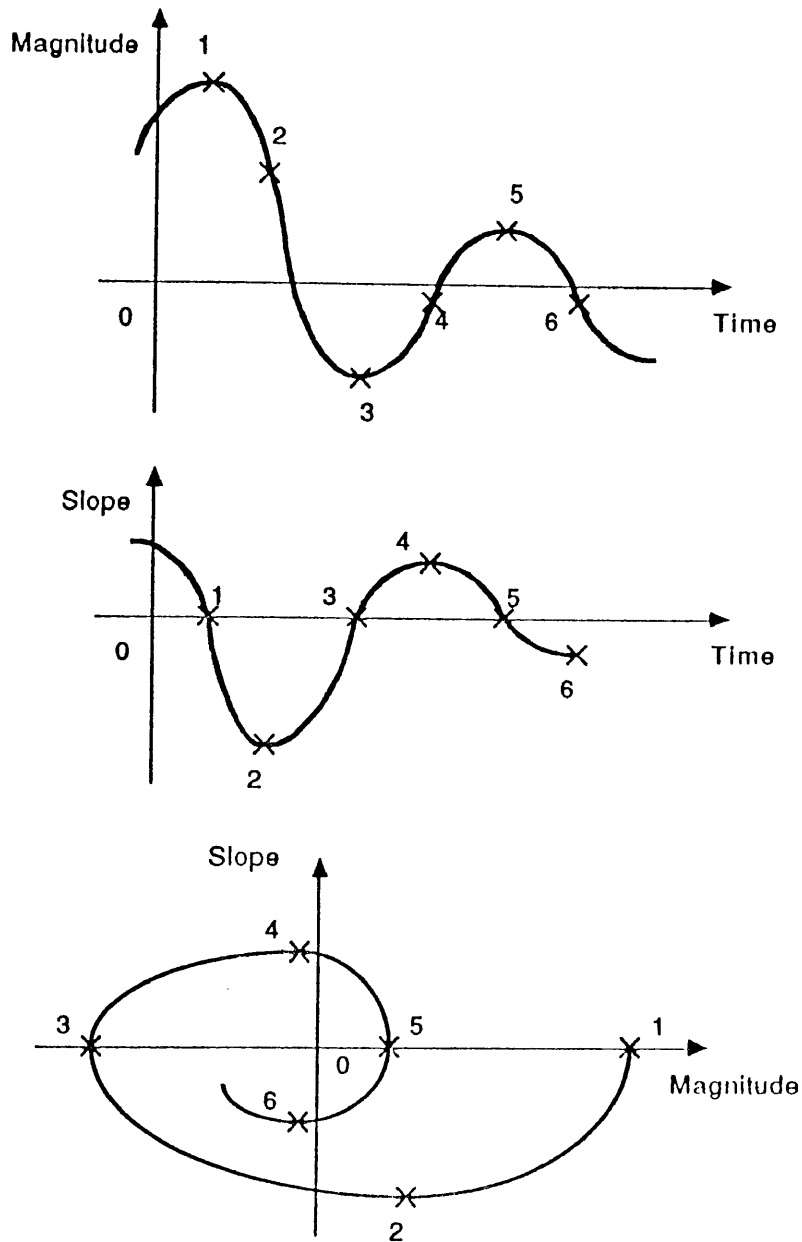


Fig. 18: Signal Locus in the Magnitude-Slope Plane

When the input signal is periodic, the locus is a closed curve. In the case of a symmetrical, high-frequency ramp, two situations occur. During the rising slope of the waveform, the derivative is constant and positive ($2Af$). During the falling slope, the derivative is also constant, but negative ($-2Af$). The locus corresponding to such a waveform is a rectangle, centered around the horizontal axis as shown in figure 19. For non-symmetrical ramps, the rectangle changes position (figure 20).

A sinusoidal input signal can be written as

$$x(t) = x_{off} + A \sin(\omega t + \phi) \quad (54)$$

With x_{off} the offset, A the amplitude, ω the angular frequency and ϕ the initial phase angle.

The derivative of the signal is given by

$$x'(t) = A\omega \cos(\omega t + \phi) \quad (55)$$

It is clear that the locus of such signal in the two-dimensional plane will be an ellipse of width A and of height ωA . The ellipse will always be centered around the horizontal axis (figure 21). A change in amplitude will widen it, a change in frequency will stretch it vertically and a change in offset will shift it horizontally. A phase shift will not affect the position of the locus, since the same points occur, only at different times.

From these two examples, it is clear that depending on the waveform, the frequency and the offset of the periodic test signal that is applied to the ADC, different loci will be executed in the signal plane. Dynamic testing of the ADC will consist in trying to cover a section of the signal plane sufficiently well, using such loci, in order to find the exact dependence of the INL values upon x' , as well as x .

When an input waveform with known locus in the signal plane is sampled at specific instants by an ADC, it is clear that each sample can be represented as exactly one point in the signal plane. Obviously each such point is located somewhere on the signal locus. When the ADC operates at a constant sampling rate, the horizontal distance between two consecutive points will be roughly

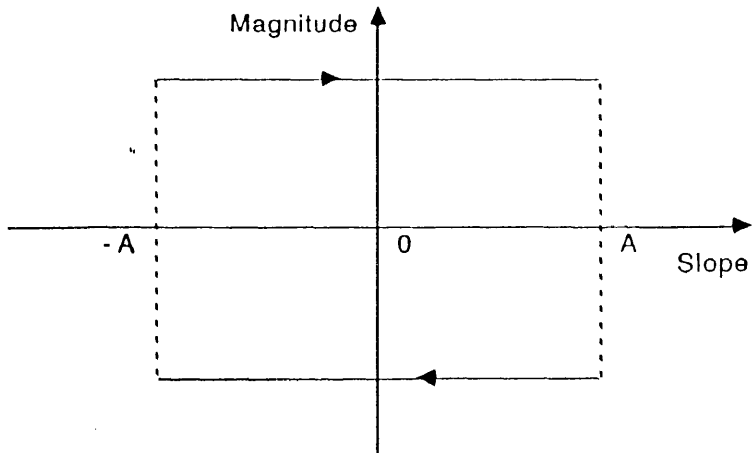
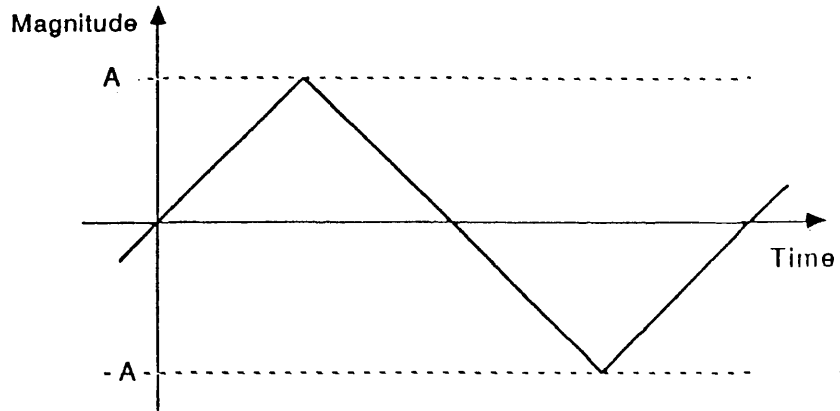


Fig. 19: Locus of a Ramp

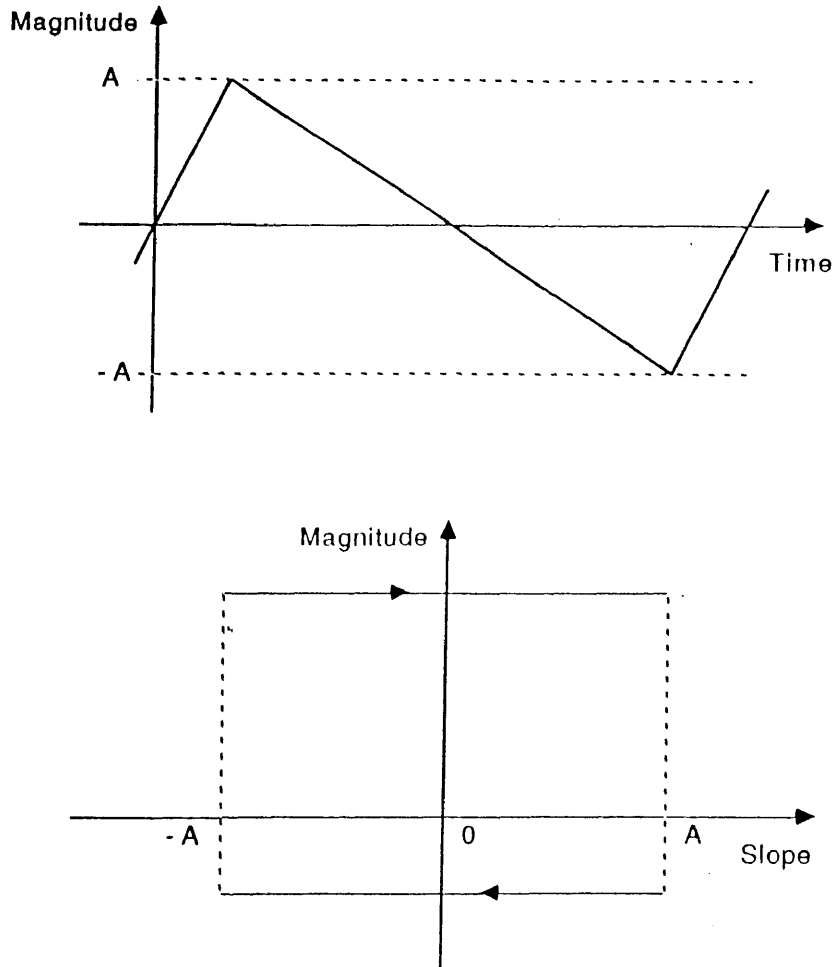


Fig. 20: Locus of an Asymmetric Ramp

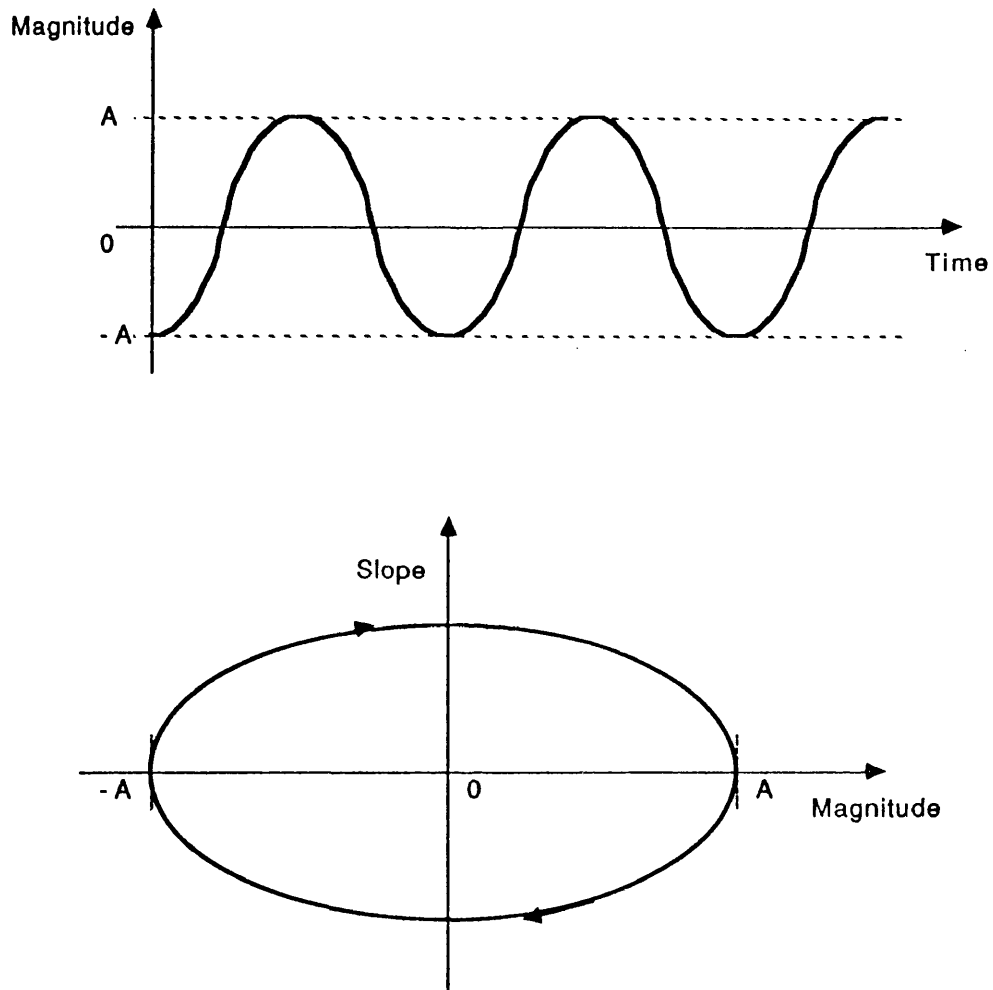


Fig. 21: Ellipsoidal Locus

proportional to the slope of the signal (first derivative). The vertical distance will be roughly proportional to the second derivative of the signal (figure 22).

3.3. Dynamic Histogram Test

It is clear that when a histogram test is performed using a high-frequency waveform, the influence of the derivative of the signal should be taken into account. This necessitates a generalization of the histogram concept. In the classic sense, a histogram is represented as a two-dimensional diagram, on which one dimension represents the bin number and the other dimension a count (tally) of values. Both dimensions are represented by integers, since there is a finite number of bins (2^N) and a finite number of samples in each bin. The number of samples in each bin is normalized relying on the probability distribution of the input signal in order to obtain normalized bin widths (in lsb). These bin widths are normally represented by *real* values, from which DNL and INL can easily be derived.

In this traditional histogram, the bin number is roughly proportional to the value of the input signal (at least for nearly ideal converters). However, no provision is made to differentiate between different signal slopes (derivatives) that could be associated with each bin. To include this second variable, a second dimension needs to be added to the histogram domain, so as to also reflect the signal slope. A problem is that the slope, unlike the bin number, is a continuous and not a discrete quantity. In order to make a histogram, one must have a finite, discrete domain, since the purpose is to tally samples falling into different sections of that domain.

The two-dimensional histogram domain will be represented as finite, discrete regions in the magnitude-slope plane. One coordinate of each region will correspond to one specific set of signal magnitudes. In practice, the X coordinate will be the bin number, corresponding to all signal magnitudes yielding a specific ADC output code at DC. The Y coordinate will also be discretized. This can be accomplished by dividing the range of possible slope values into several (arbitrarily chosen) sections. One elementary region in the histogram domain will then correspond to a rectangle in the (continuous) signal plane. In particular, it will correspond to a set of possible signal values, of which the magnitude is such that at DC, they would yield the same ADC output code (same ADC bin).

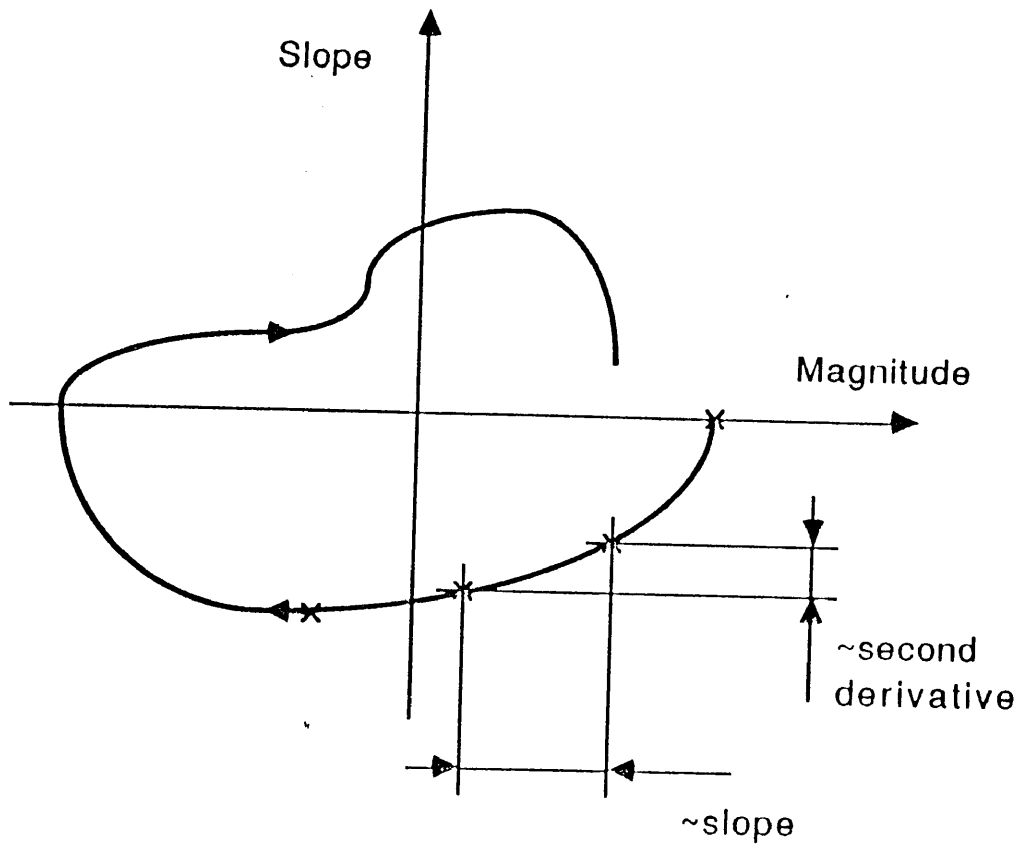


Fig. 22: Relationship between Samples and Locus

while simultaneously the slope would be within certain pre-established limits, like depicted on figure 23.

One could wonder how many slope intervals need to be considered. In practice, it is better not to consider too many (definitely not 2^N), since the size of the histogram and the number of samples required to define it, would quickly become prohibitive. The scale of the slope axis really depends on the desired accuracy, the expected frequency range of the signal and the sensitivity of the ADC transfer curve to signal slope variations. It should be noted that the slope of a signal can be positive or negative, and the slope axis extends above and below the magnitude (bin number) axis. The progression of different slope values along that axis is essentially irrelevant. The scale could be linear, but in some applications it may be preferable to choose a (quasi-) logarithmic reference system.

Generalized (or dynamic) histogram testing boils down to counting the number of samples falling into each discrete section of the magnitude/slope plane (each region of the histogram plane), when the shape of the input signal is known. The counts (tallies) can be graphed as a third (vertical) dimension, like is shown in figure 24. By applying different input signals, one could theoretically determine tallies for each point in the histogram domain. However, some precautions must be taken when different measurements are combined onto the same graph.

For any particular kind of periodic input signal (e.g. a sine wave of a certain frequency and amplitude), a locus of magnitude-slope points is described in the signal plane. A corresponding, discrete locus is described in the histogram domain, since to each sample taken, there corresponds one discrete point in the signal plane, which also corresponds to one region in the histogram domain (figure 23).

However, several samples (several points in the signal plane) could potentially have magnitudes that correspond to the same bin (same X value in the histogram plane), but slope values that correspond to different Y values in the histogram plane (figure 25). Something similar is possible when the input signal is complex (e.g. superposition of sine waves of different frequencies). Such a situation is undesirable when a dynamic histogram test is performed. The test procedure is simplest when the discrete locus in the histogram domain is be such that to each magnitude point would correspond exactly one positive slope point and one negative slope point, like depicted on figure 26.

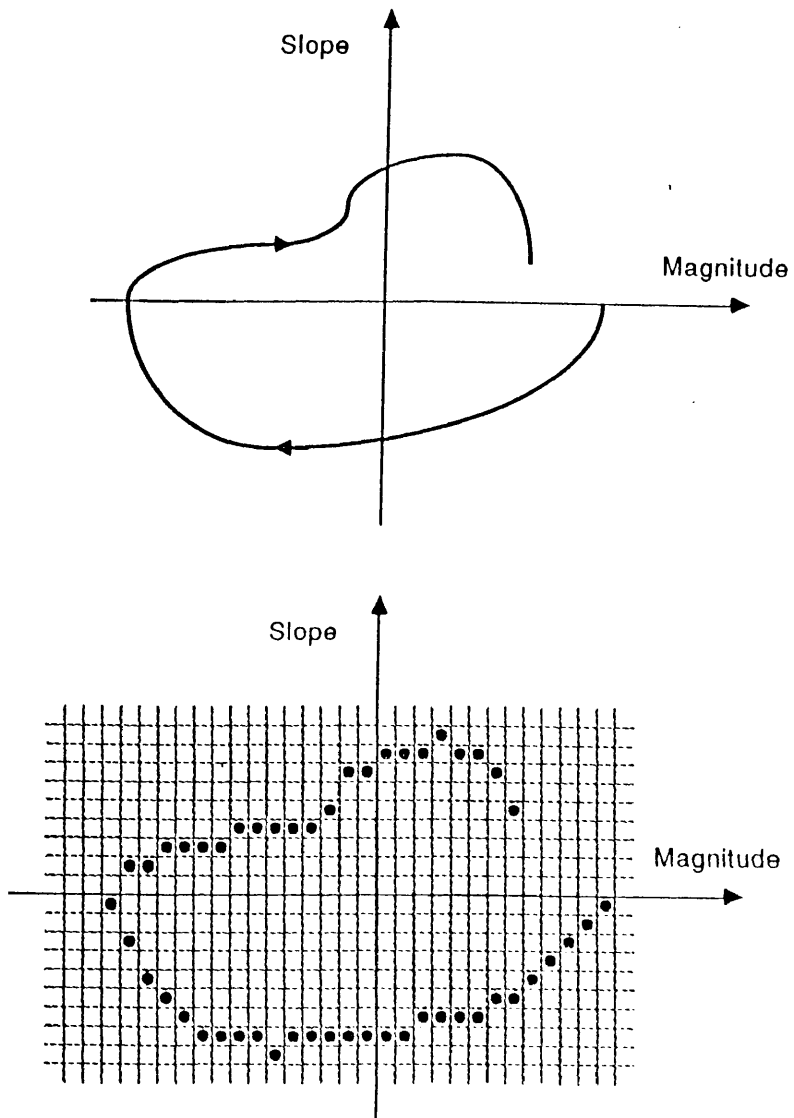


Fig. 23: Continuous and Discrete Histogram Domain

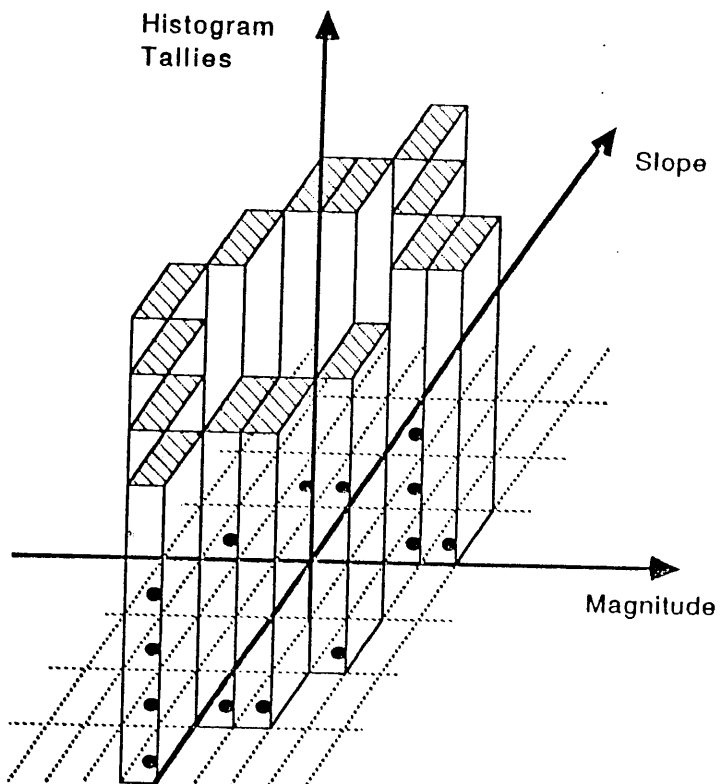


Fig. 24: Three-Dimensional Histogram Plot

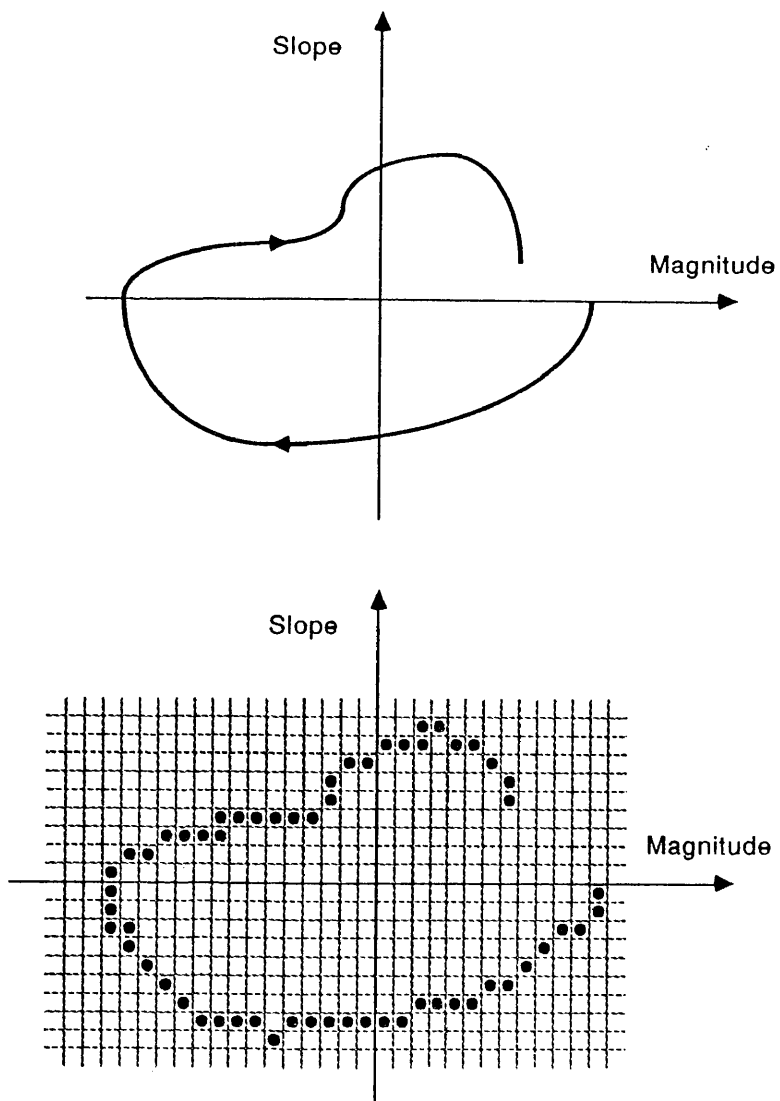


Fig. 25: Discrete Locus with More Than One Slope per Magnitude

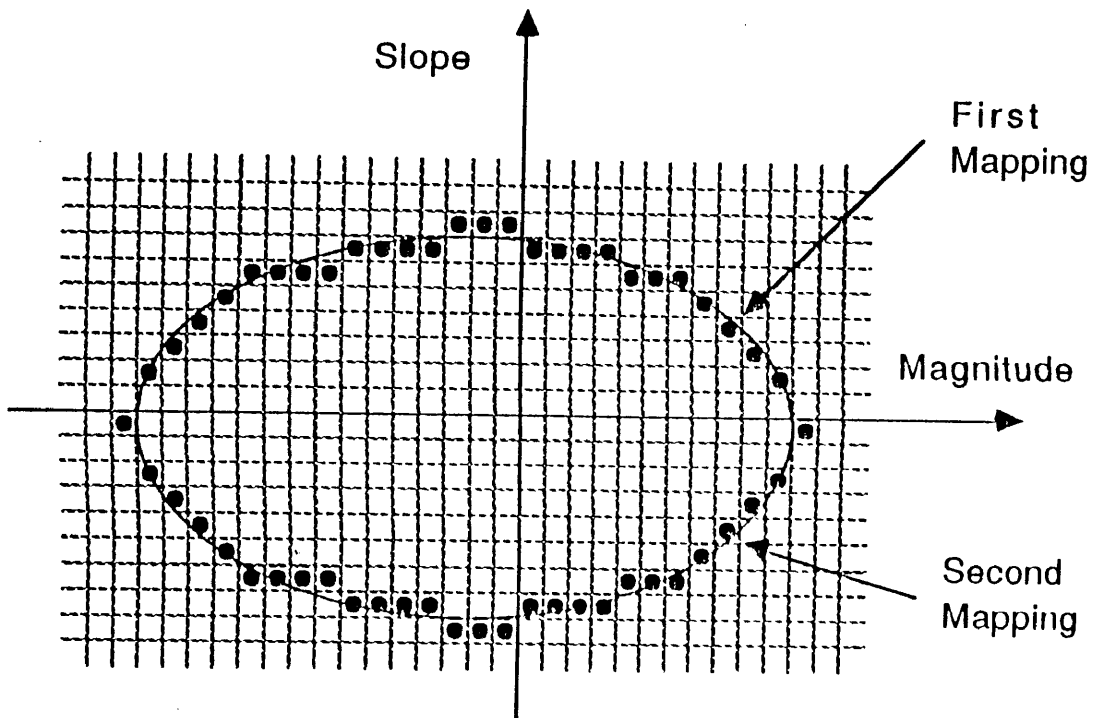


Fig. 26: One to One Mapping

When analyzing the ADC output codes during the process of performing a histogram test, it is obvious how to allocate each sample to a particular magnitude (bin number). However, it may be harder to determine in what section of the slope axis the sample falls. In order to avoid having to use any external measurements, the slope values can be expressed in output-referred rather than absolute units, in this case *lsb* per sampling period, rather than volts or amps per second. The slope of each sample can be estimated by comparing it with the previous sample(s) (numerical differentiation). If needed, the values could later be converted to absolute (input-referred) values if the input signal range and sampling frequency are known.

A methodical way to perform a dynamic histogram test is the following. First, the maximum signal amplitude and frequency to be applied to the system is chosen. From that information, the maximum positive and negative slope of the signal is determined. Initially, we will assume that only sine waves are applied. The maximum slope of a sine wave of frequency f and amplitude A is $\pm 2\pi fA$. This value can be converted to output-referred units when the number of bits N and sampling rate f_s of the ADC are known, by multiplication by $2^N/(A f_s)$. The maximum slope is then $\pm 2\pi 2^N f/f_s$, in units of *lsb*/sample.

It is worth noting that the factor f/f_s represents the normalized sampling frequency, in signal periods per sample. Its inverse, f_s/f represents the normalized signal period, in samples per signal period. This quantity is very important in signal analysis. For instance, f_s/f should always be larger than 2 to satisfy Nyquist's sampling criterion.

Next, the slope axis is defined, by dividing the total range of possible slopes (both positive and negative) into a number of discrete intervals. This can be done according to either a linear or a logarithmic scale.

Next, a mapping is defined between amplitudes and slopes. This mapping reflects the locus that is normally described by the input signal in the discrete histogram (bottom) plane. The purpose is to simplify the process of assigning sample counts (tallies) to the different histogram plane regions when the histogram is actually performed. A straight-forward way to do this would be to take an amplitude equal to the mid-point of each bin, and to calculate the corresponding slope (actually, there are two mappings: one for positive slope and one for negative

slopes). When a linear scale is used for the slope values, the mapping will reflect an elliptic signal locus, made up of discrete points. If a logarithmic scale is used, the ellipse will be stretched vertically in a way as to resemble a rectangle.

The magnitude a of the input signal can be written as

$$a = a_{off} + A \sin(2\pi t + \phi) \quad (56)$$

The slope s can be written as

$$s = 2\pi f A \cos(2\pi t + \phi) \quad (57)$$

This slope can be expressed as a function of the magnitude:

$$s = \pm 2\pi f A \sqrt{1 - \frac{a - a_{off}}{A}} \quad (58)$$

This can be further normalized to output-referred units, by multiplication by $2^N/(A f_s)$ *lsb*. Alternatively, the calculation could be performed immediately in the correct units, *lsb* and *periods per sample*. Once the nominal slopes corresponding to each magnitude bin (center point) are defined, the associated slope segments are determined (it is assumed that there are less slope segments than magnitude segments or bins).

Now, the histogram can be constructed. The input signal is applied to the ADC, and output codes are tallied. Each tally is assigned to a particular region in the histogram plane, according to the magnitude/slope mapping. Since to each magnitude could correspond *two* slope values, some elementary analysis of the output data record will have to determine if a sample is part of the rising slope or the falling slope of the signal. It is very important *not* to combine both tallies, since a lot of dynamic ADC errors tend to be roughly proportional to the input signal slope, and would mask each other out if the tallies of a positive and a negative slope corresponding to the same magnitude, were to be combined.

Finally, after a complete histogram has been derived from the data record, the tallies can be used to calculate normalized bin widths, DNL and INL. It has to be noted that these operations will have to be performed for *two* histograms: one corresponding to positive slopes, the other one corresponding to negative slopes.

For the purpose of calculating bin widths, abstraction is temporarily made of the slopes, and the traditional (one-dimensional) histogram procedure is followed. If sine waves are used, the histogram tallies will have to be normalized to compensate for the non-uniform magnitude distribution, as described earlier (signal magnitude and offset will have to be determined very precisely, using an iterative fitting algorithm on the output data record).

Obviously, the histogram points corresponding to the smallest and to the largest signal magnitude will belong to both histograms at the same time, since the slope is nominally zero. This is of no concern, since the first and last bin of a histogram test are discarded anyway. It has been shown above that this procedure to derive the ADC transfer characteristics from a histogram, corresponds to an end-point normalization, in the sense that by definition, the first and the last code transition levels (ctl_1 and ctl_{2^N-1}) are ideal.

This makes it possible to easily combine the results for the two histograms of the same sine wave, since the first and the last bin of the two histograms correspond to the *same nominal magnitude*, as well as the *same nominal slope* (zero). For continuity reasons, the two sets of bin widths, DNL and INL values will be scaled in exactly the same way, and thus be compatible.

This fact makes it more attractive to use sine waves for dynamic histogram testing rather than ramps (despite the more complex normalization procedure for sine waves). If testing were done with a ramp, all magnitude values in the histogram plane could be covered, but only two slope values (one positive and one negative) would occur. This would result in two essentially independent histograms, one taken for the positive slope and the other one taken for the negative slope. Since the two histograms have no points in common, they would end up being normalized (bin width calculation) independently. The DNL and INL values would be normalized independently as well and any offset or gain error associated with one slope and not the other would not be accounted for.

3.4. Full Characterization

The power of the generalized histogram becomes apparent when more than one measurement is combined on the same graph. Instead of applying only one sinusoidal input signal, one could apply several different ones, calculate the

respective normalized bin widths, and display them as part of one system response to a two-dimensional excitation space (the bottom plane of the histogram).

Different measurements will be compatible with each other when sine waves of constant magnitude and offset but different frequencies are considered, since all the bin widths (after normalization) will be normalized in the same way. This is because the first and last bin of each histogram corresponds to the same values of magnitude and slope, as mentioned above. All elliptic loci in the histogram plane are concentric, with common left-most and right-most point. If amplitude or offset are changed from one measurement to another, the loci will not have their extremes in common anymore. If the ADC is non-ideal for small or large input signals, one *lsb* may not represent the same analog quantity for the different measurements, and the normalized bin widths may be incompatible.

By taking different measurements using sine waves of different frequencies, computing the respective histograms and normalizing them in order to obtain normalized bin widths, it is possible to cover all the points within a certain section of the histogram plane. Once this information is available, the ADC response can be accurately predicted for *any* kind of dynamic input signal. In other words: complete dynamic characterization of the ADC is possible using the dynamic histogram method.

One can calculate the effective code transition levels of the ADC for any combination of magnitude and slope at the input. In order to do this, the dynamic bin widths have to be reorganized. Instead of considering bin widths along one particular signal locus (ellipse), one should first consider all experimentally determined bin widths along the same horizontal line (the bin widths for a certain slope). This is shown in figure 27.

Ideally, the sum of all the bin widths should be $2^N - 2$ (first and last bin are discarded). Due to measurement error (finite number of samples in the histogram etc.), this may not be exactly the case, and normalization along the constant slope line may be necessary. This normalization is accomplished by dividing each bin width by the sum of bin widths along the line, and multiplying by $2^N - 2$. After this operation, DNL values for that particular slope can be obtained by subtracting 1 from each value. INL values can be obtained through a partial summation of DNL values, and code transition levels (in *lsb*) by adding n (bin

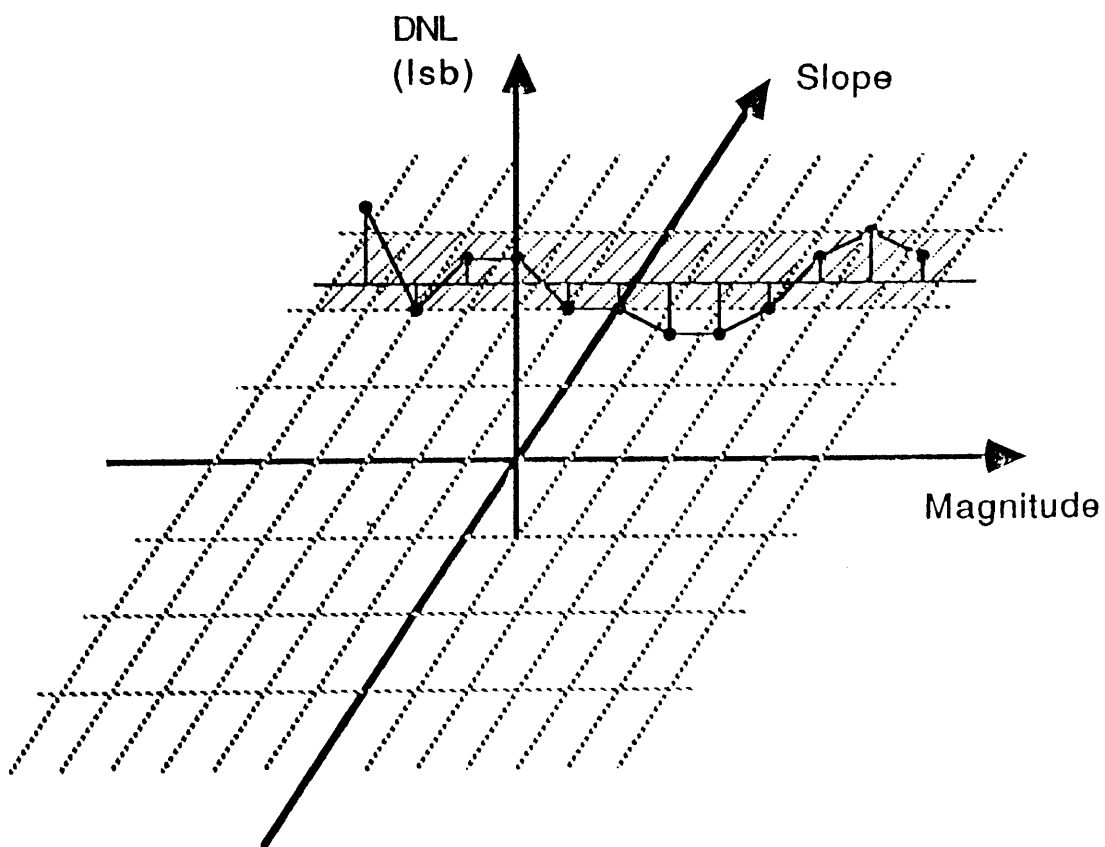


Fig. 27: DNL Values along a Horizontal Line

number) to each INL value.

To calculate the response to an arbitrary dynamic input signal, the locus of the signal needs to be determined in the histogram plane (discrete magnitude/slope plane). Magnitude and slope of the signal should be expressed in lsb and lsb per sample respectively. If the signal is periodic, the locus will be a closed curve. If not, an arbitrary, open curve will be described. Next, the magnitude/slope values of the input signal should be determined at each sampling instant, and the corresponding point in the histogram plane should be determined. Along the line of constant slope that contains each point of the signal, the transition level immediately below the given magnitude should be found. The bin number associated with this level will reflect the output code (figure 28).

Of course, the procedure is rather involved, and if carried out completely, the use of specialized computer programs would be almost mandatory. However, the main use of the generalized histogram test and associated complete dynamic characterization of the ADC is not to calculate the dynamic ADC response to arbitrary input signals. Instead, it could provide a powerful tool to classify dynamic errors and analyze their origin within the ADC, in a way similar to the analysis of static errors.

A limitation of this particular approach, is that only magnitude and slope of the input signal were considered significant variables, and not higher derivatives. Including higher derivatives would significantly complicate the procedure (the histogram plane would be three-or more-dimensional). At the same time, it appears that many common errors mainly depend on the first derivative of the signal, and as such the proposed scheme is sufficient for diagnostic purposes.

As an example, figure 29 shows the effect of a magnitude-dependent timing error in an ADC, and the associated error when sampling an asymmetric ramp signal. Magnitude-dependent timing errors are very common dynamic errors in extremely high-speed flash converters (discussed in chapter IV). They could be caused by different parasitic delays (e.g. due to distributed capacitance) in clock lines, or capacitive mismatches in digital latches associated with a number of comparators that are physically located far away from each other on a chip.

Such errors cannot be detected by a static (low-frequency) histogram test.

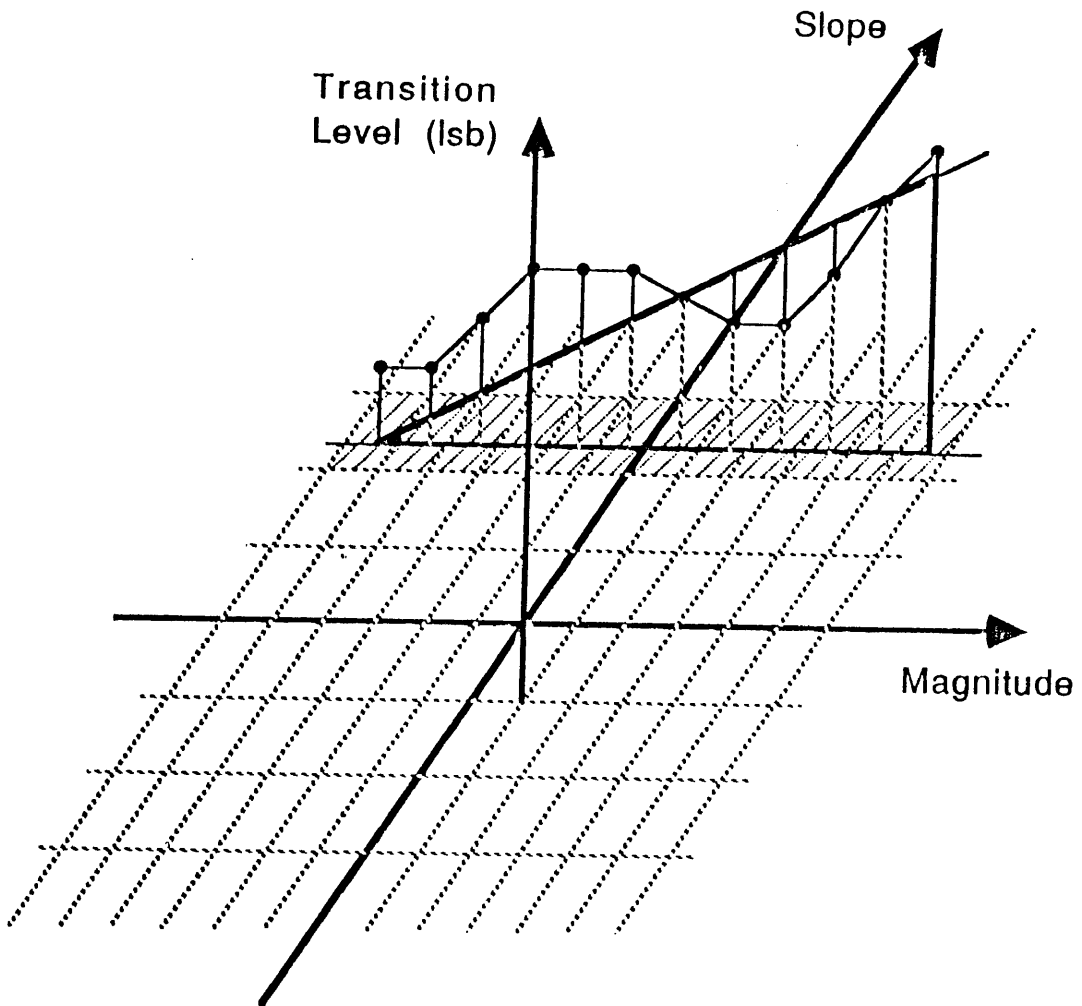


Fig. 28: Transition Levels along a Horizontal Line

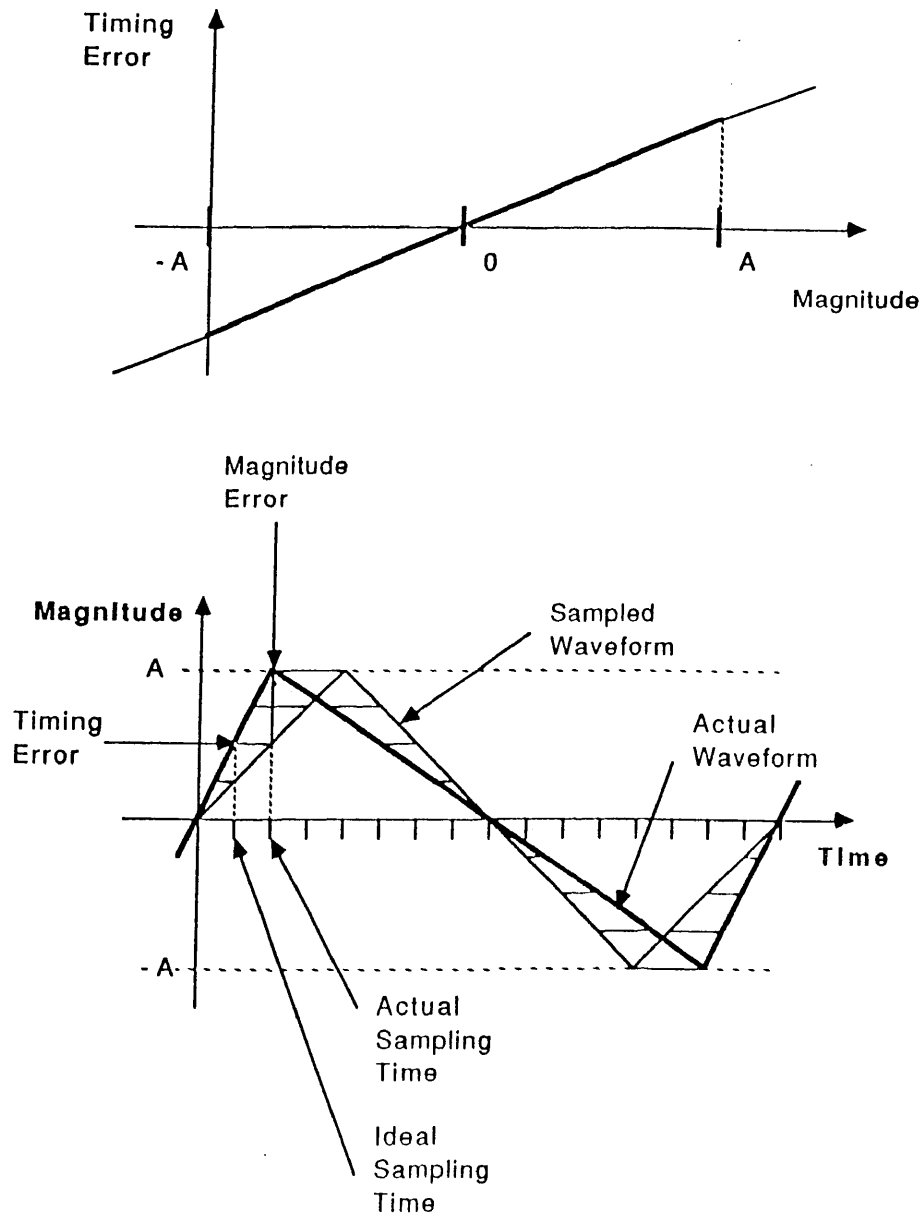


Fig. 29: Magnitude-Dependent Timing Error

A high-frequency histogram test may partially or completely hide the errors, since if the input waveform were symmetrical, as many positive as negative errors would be made, with no net result on the histogram tallies. The distortion can be detected using the FFT method, but precise diagnosis is usually impossible. Dynamic histogram testing would make it possible to precisely reconstruct the delay pattern from the known slope and associated variation in transition level over the different magnitudes. Although further research is suggested in this area, it seems likely that other common dynamic effects like limited frequency response, limited speed of internal amplifiers or hysteresis could be classified and diagnosed correctly through dynamic histogram testing.

3.5. Conclusion

It was shown how high-frequency input signals can corrupt the readings of a traditional histogram test. This effect was attributed to the derivatives of the input signal exciting dynamic error mechanisms within an ADC. It was shown that the influence of the first derivative (the slope) is dominant. As a result, higher derivatives can be neglected, and an input signal can be represented as a locus of points in a two-dimensional magnitude-slope plane. This plane can be discretized into a "histogram" domain plane, which can be used as a base for an extended, dynamic histogram test. The procedure for performing this test using sinusoidal input signals was described. Finally, it was shown how the results of such a test could be used as a full, dynamic model for the ADC, in order to calculate the response to an arbitrary, high-frequency input signal.

CHAPTER IV

ANALOG TO DIGITAL CONVERTER ARCHITECTURES

4.1. Introduction

After a general discussion about analog to digital converter transfer characteristics and testing, this chapter introduces some particular hardware configurations that are commonly used to perform the analog to digital conversion. Some architectures, (in particular the multi-step and pipelined converters) will be treated in more detail later. Errors in their transfer curve will be related to particular elements of the hardware, and techniques will be introduced to compensate for these effects.

4.2. Operation of a Flash Converter

The most straightforward way to implement an A/D converter is probably according to the "flash" principle. Due to its inherent simplicity, this type of converter can be made extremely fast, and impressive results have been reported in literature [12, 13, 14, 15, 16, 17, 18, 10, 19]. In a flash converter, the input signal is connected to a bank of fast comparators (figure 30), of which the reference voltages are chosen so as to coincide with the desired code transition levels of the converter. These reference voltages are usually derived from a single reference level through a resistive divider.

It is clear that in order to obtain a converter with an effective resolution of N bits (2^N possible output codes), $2^N - 1$ comparators will be needed, since the input voltage range must be divided into 2^N bins. As a result, there will be $2^N - 1$ digital comparator outputs, which represent the input signal in "thermometer code" format (the first code is $00 \cdots 0$, the second one is $10 \cdots 0$, the last one $11 \cdots 1$). The number of output lines (and of course IC pins) quickly becomes prohibitive for increasing resolutions. To avoid this problem, a digital encoder is normally used, which converts the $2^N - 1$ bit thermometer code into N bit straight binary coding.

The input signal is usually buffered before being applied to the comparator inputs. The buffer can be a simple amplifier, in which case the comparator inputs

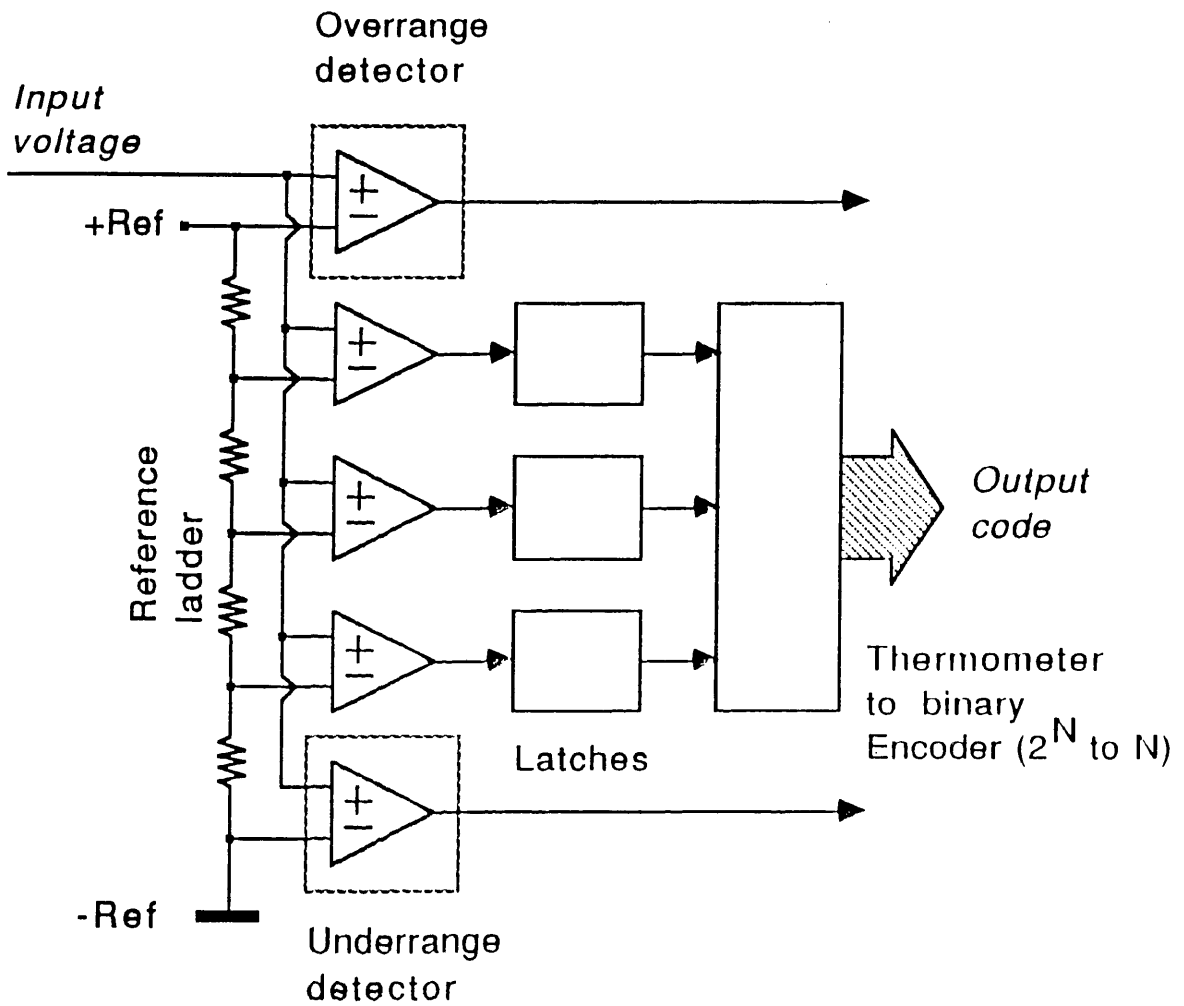


Fig. 30: Flash Converter

may be changing while the comparison is being performed. The comparators outputs are usually passed on to digital latches, which will sample the outputs when a timing signal is applied. It is also possible that the input buffer be realized as a sample/hold amplifier, which will freeze the input signal while the comparison is being performed. Ultra-fast flash converters usually avoid using a sample/hold amplifier, since this tends to limit the maximum allowable bandwidth of the input signal.

4.3. Multi-Step Converters

The main disadvantage of the elementary flash converter is the large component count (and hence chip area and power consumption) required to realize higher resolution schemes. The number of comparator/latch combinations ($2^N - 1$) quickly becomes prohibitive for resolutions beyond 7 or 8 bits. In order to circumvent this problem, multi-step architectures have been developed [20, 21, 22, 23, 24, 25, 26].

A multi-step converter divides the useful input signal range into a number of sub-ranges using a comparator bank, like in the flash converter (for this reason, we will often refer to this section of the circuitry as the flash section). The difference, compared to the full flash converter, is that the number of sub-ranges is less than the total number of desired bins (2^N for an N bit converter).

Next, the comparator output codes are used to address a bank of voltages (or currents, depending on the converter type) to be subtracted from the input signal (figure 31). The effect of this operation is to calculate an analog "residue" or remainder, which reflects the portion of the input signal that exceeds the lower boundary of the signal section in question (figure 32). Naturally, the residue of the conversion will have a range that is smaller than the range of the input signal. It will normally never get larger in value than the range of one section.

The bank of fixed values that are subtracted from the input signal in order to calculate the residue of one stage, can be seen as an elementary digital to analog converter (DAC). Indeed, the input to this section is a number of *digital* lines (the thermometer-coded outputs from the latches), while the output of that section is an *analog* value, to be subtracted from the analog input signal. Throughout the remainder of this text, we will often refer to this section as the "DAC" section of

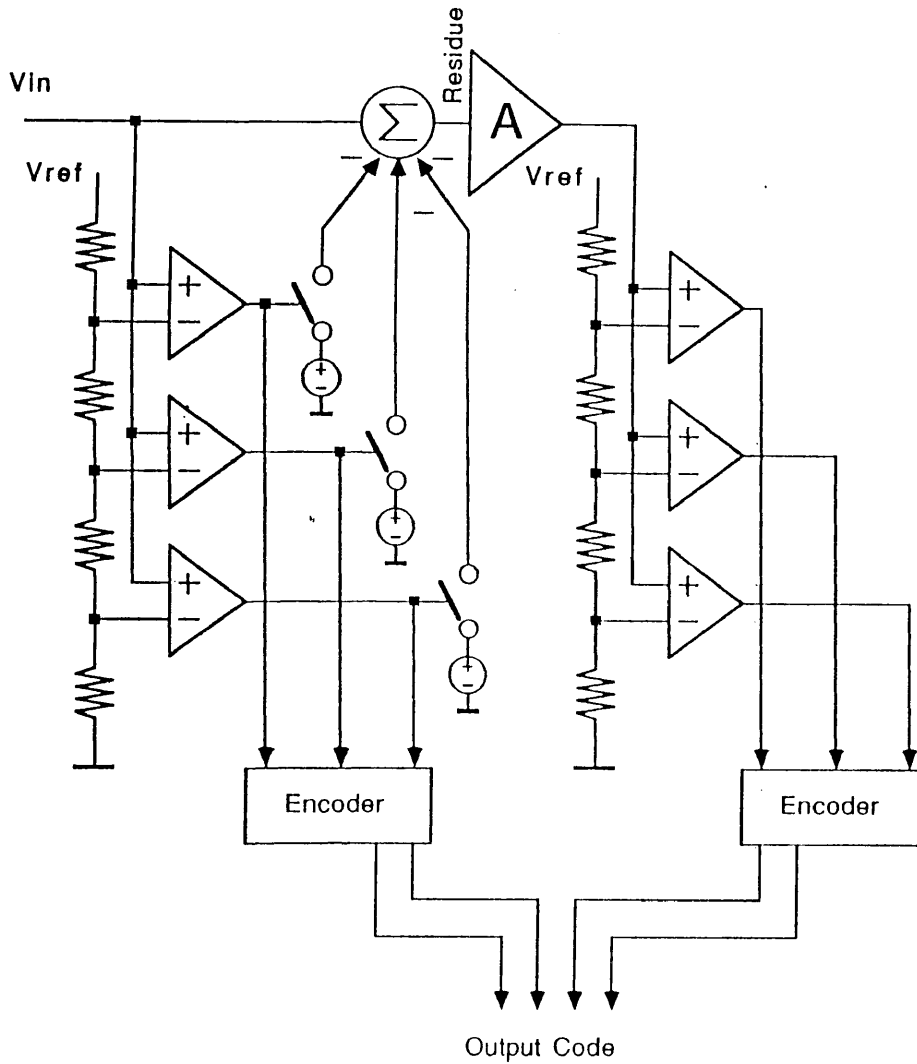
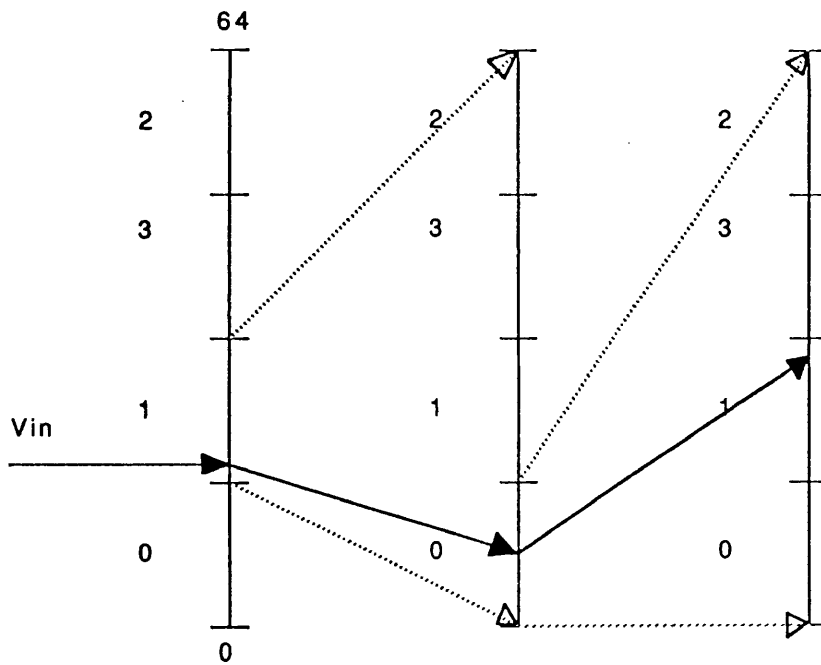


Fig. 31: Multi-Step Converter



$$\text{Output} = 1 * 16 + 0 * 4 + 1 * 1 = 17$$

Fig. 32: Sub-Ranges

a converter stage. The comparator bank with latches and reference circuitry will be referred to as the "flash" section, for obvious reasons.

The second step of the conversion consists in taking the residue from the first step, and converting it in way similar to the first step. If the circuitry implementing the second step (the second "stage") does not have a gain stage at its input, the range of the signal to be converted will be significantly smaller than the range of the first stage. As a result, the comparator reference levels must be scaled accordingly. This configuration is often referred to as a "sub-ranging" converter. However, if the second stage has an input amplifier that amplifies the signal to the same nominal range as the input signal of the previous stage, the same comparator reference levels could be used. This situation is often preferable, since a multi-step converter can be implemented using a number of nominally identical stages, cascaded behind each other.

A common strategy for multi-step converters is to design the first stage so that the comparator levels of that stage would divide the input signal range into 2^M nominally identical sections (bins). One can easily verify that after thermometer-to-binary encoding, the comparator outputs of the first stage will express the M most significant bits of the overall conversion result. The nominal range of the residue of the first stage will be $R/2^M$. This residue can be forced to the same range as the input through multiplication by 2^M . If the second stage is similar to the first one, its (binary-coded) output will reflect the next M bits of the overall conversion result. Additional stages can be cascaded in order to improve the resolution of the converter (figure 33). However, it will be shown in subsequent chapters that this is not the only viable strategy. It is possible to design converter stages that have a gain that is not a power of 2, or not even an integer value.

It may seem that cascading identical stages in a multi-step architecture would be an ideal mechanism in order to realize extremely high resolution converters, since it would suffice to cascade more (identical) stages. It is true that increasing resolution would be obtained, but overall accuracy will be limited by the unpredictability some of the elements: mainly the accuracy of the comparator reference levels, the precise value of the subtracted voltages (DAC levels) and the value of the interstage gain. The effect of these inaccuracies will be described in great detail below, since they are the primary focus of the research described in

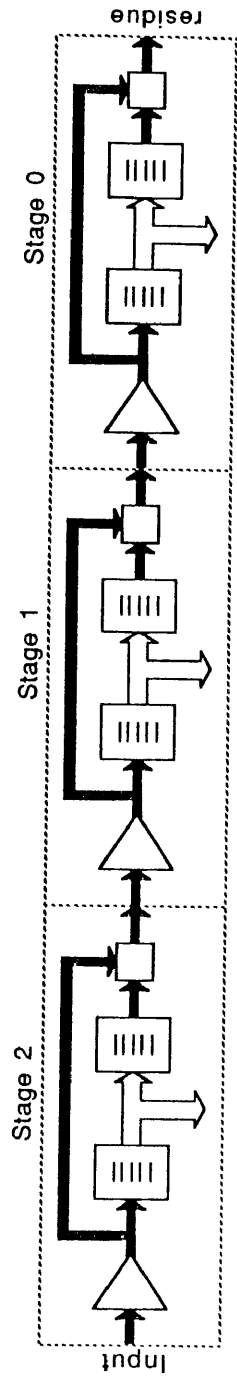


Fig. 33: Cascaded Converter Stages

this dissertation.

4.4. Pipelined Converters

Pipelined A/D converters are a special case of multi-step converters. They have gotten their name not because of the particular conversion algorithm involved, but because of the organization of the different stages. The pipelined approach has been used extensively for medium to high speed (a few MHz), medium resolution (10-13 bits) converters, especially in CMOS or BiCMOS technologies [27, 28, 29, 30, 31, 32, 33, 23, 24, 34, 35,36, 37, 38, 39, 40, 41].

In a pipelined converter, each stage is preceded by a sample/hold amplifier, which samples the input signal and then holds it constant for a certain time, while the conversion is being performed. After the comparator outputs are latched, a voltage determined by the comparator outputs is subtracted from the signal and a residue is calculated (DAC operation). This residue is then sampled by the sample/hold amplifier of the next stage, and held for the time of its conversion.

The sequence of operations in each stage is synchronized by a multi-phase clock. The clocking scheme is determined in such a way as to realize a pipeline, in the same sense as pipelined logic is organized in digital circuits. Such operation requires that the input of a stage would be sampled during one clock phase, while the output becomes available during a different clock phase. When the clocking is designed so that the input phase of the next stage would coincide with the output phase of the previous one, continuous operation of several cascaded stages is possible.

It is well-known that an input signal can be applied to the first stage in line, as soon as the second stage has sampled the output of the first one. As a result, it is not necessary to wait for a signal to travel through the whole pipeline before applying the next signal. The overall throughput can be increased significantly as compared to the time it would normally take the analog signal to travel through the complete pipeline (figure 34).

All stages in a pipelined converter are nominally identical, and all of them have an input sample/hold amplifier in order to make the pipelined operation possible. However, in practice there exists a difference between the first stage and the other ones. The sample/hold amplifier of the first stage must be able

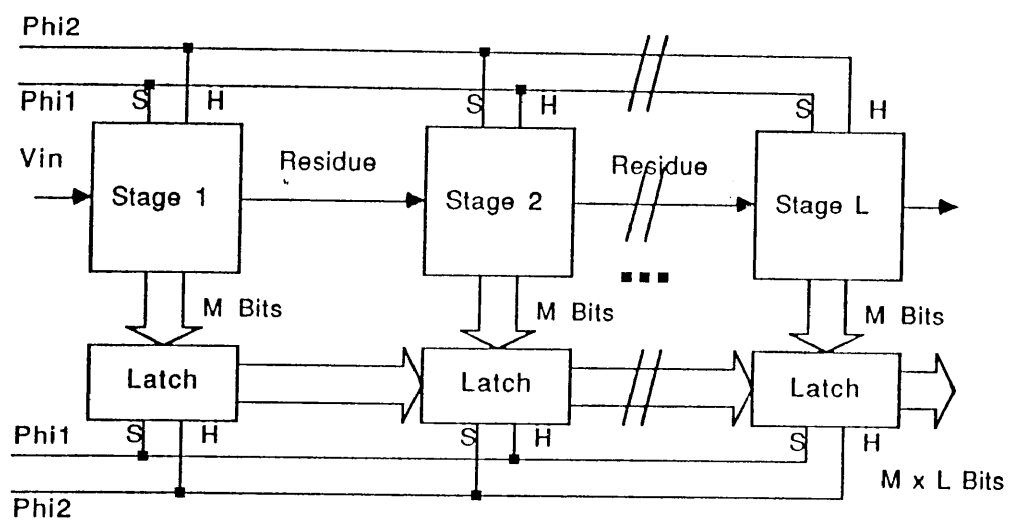


Fig. 34: Pipelined A/D Converter

to sample a rapidly changing external input signal without noticeable distortion, while the sample/hold amplifiers of the other stages only see the DC output level (residue) of the previous stage. As a result, the design of the first stage usually requires extra precaution.

Although the pipelined multi-step configuration that was just described is probably the most common one, it is not the only possible one. Other conversion algorithms have been implemented in a pipelined fashion. Pipelined converters which implement a successive approximation-like search have been reported [42, 43].

4.5. Successive Approximation Converters

Although successive approximation converters are not the primary focus of this research, they deserve a short discussion due to their popularity for achieving relatively high resolutions (13-15 bits) at limited conversion rates (100 kHz-1MHz) [44, 45, 46, 47]. They also exhibit some interesting similarities with schemes we will discuss, in particular with recycling converters.

A successive approximation converter uses a precision digital to analog converter (DAC) in order to perform the analog to digital conversion. The DAC is used as part of a binary search algorithm, which aims at applying different digital codes to the DAC until the best match for the input signal is obtained. In many cases, a binary-weighted capacitor array is used to implement a charge-redistribution DAC [44]. The search algorithm works as follows.

The input signal is sampled by a precision sample/hold amplifier and held for the duration of the whole conversion. In order to obtain conversion to the N bit level, a DAC with N bit resolution is required. First, an input code corresponding to the middle of the possible range is applied to the DAC. (It is assumed that the DAC input uses straight binary coding, and not a thermometer code). That code has a most significant bit (MSB) equal to 1 and all other bits equal to 0. Next, the DAC output is compared to the input signal (output of the sample/hold amplifier) using a precision comparator, with one bit output (figure 35). If the result of the comparison is 1, the input signal exceeds the DAC output and the MSB of the conversion is known to be 1. If the comparator output is 0, the input signal is smaller than the DAC output (equivalent of $10 \dots 0$) and the MSB of the

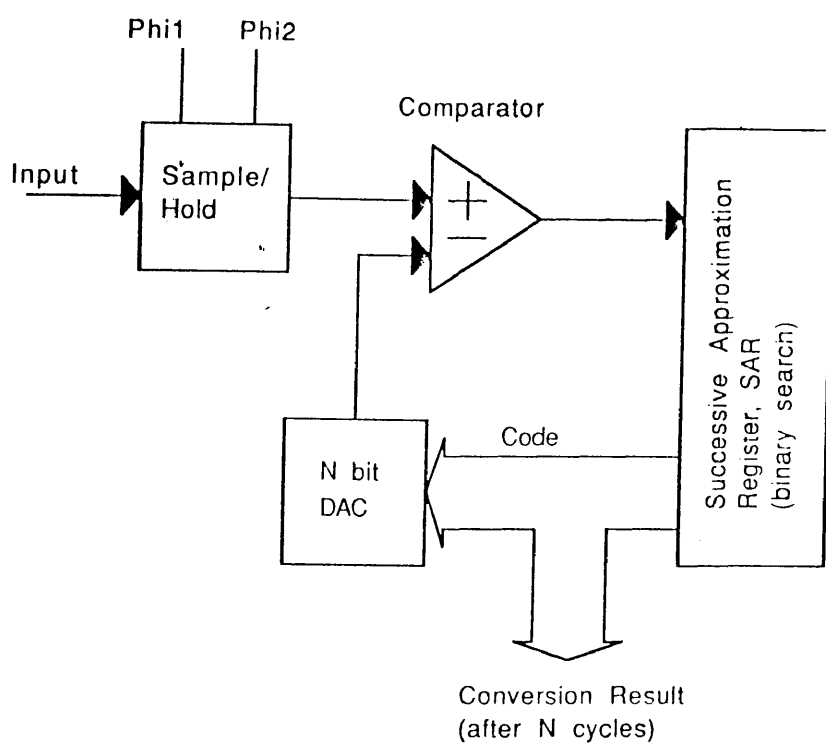


Fig. 35: Successive Approximation Converter

conversion is known to be 0.

Once the MSB is determined, a digital code consisting of this MSB, followed by a 1 and a string of 0's is applied to the DAC. The same comparator is used to determine whether the DAC output exceeds the input signal or not. This information defines the second most significant bit. The process is repeated in a similar way, using the previously determined first and second bits in order to determine the third bit etc. After N steps, the total conversion result is available and a new input signal can be sampled by the input sample/hold amplifier. The whole operation is synchronized and controlled by a special register (controller), called the successive approximation register (SAR).

An important drawback of successive approximation converters, is their relatively low conversion rate, due to the N distinct conversion cycles required for each sample. Another drawback is that a full, precision DAC must be implemented within the ADC. This can be avoided by "unfolding" the successive approximation algorithm into a multi-step or pipelined approach. In a first stage, an analog level corresponding to the middle of the nominal input signal range is compared against the input signal. If the input signal exceeds that level, it is subtracted from the input signal and passed on to the next stage, after multiplication by 2 (figure 36).

4.6. Recycling Converters

A recycling ADC is closely related to a pipelined scheme. It is another implementation of a multi-step approach, which has been used to obtain relatively high resolutions on a relatively limited silicon area [48, 36].

A recycling ADC implements the multi-step algorithm by taking an input signal, generating a local code and calculating a residue. That residue is normally fed to another stage, which can be nominally identical to the first one if the correct gain is implemented at the input.

A recycling converter realizes the same idea, using only one stage (figure 37). The output of that stage (the residue) is fed back to the input of the same stage, and the process is repeated as many times as needed to obtain the desired number of bits. Then, the input is connected to the external signal again, and the process can start over for the next sample.

It is clear that a recycling stage must have a sample/hold amplifier and be

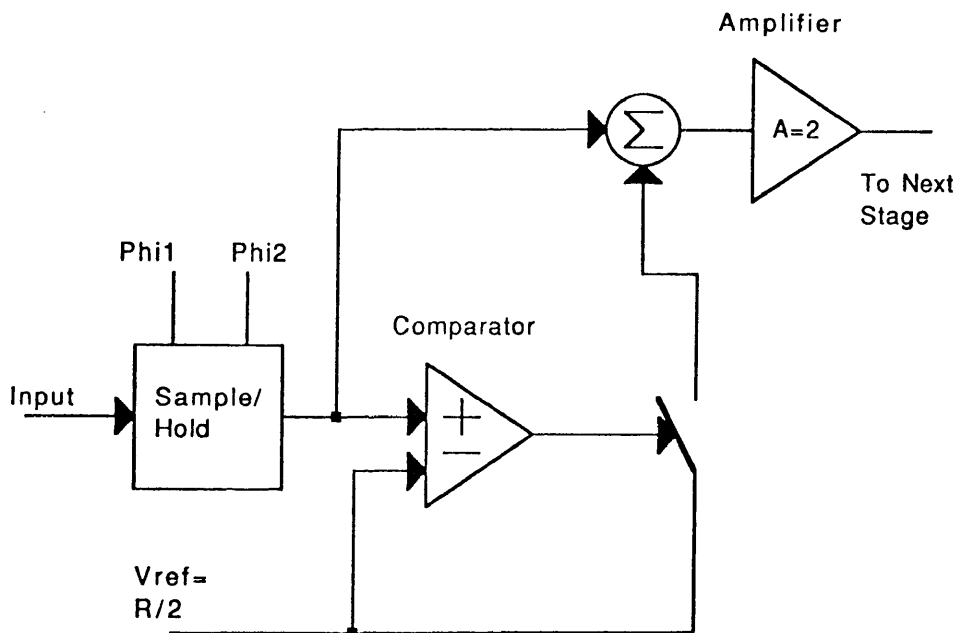


Fig. 36: Unfolded Successive Approximation Converter

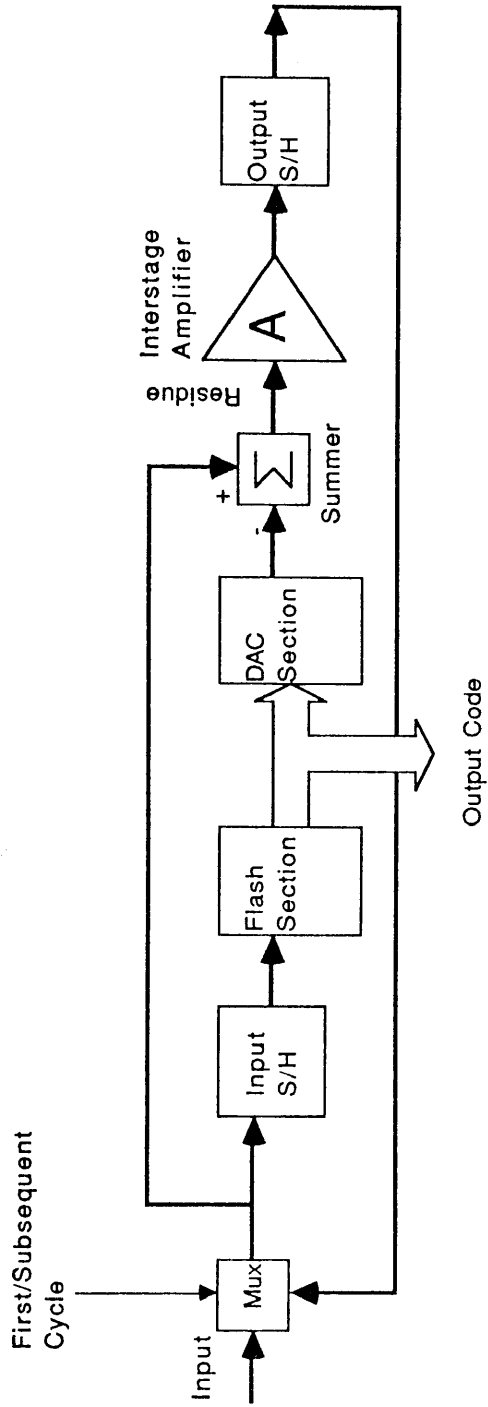


Fig. 37: Recycling Converter

clocked. just like a pipelined stage, since the output of the stage must be held until the input of the same stage is ready to sample again. If those conditions are met, a very compact, high-resolution ADC can be implemented using a single stage. The area is N/M times smaller than in a corresponding pipelined scheme (N is the total number of bits, M the number of bits in one stage). However, the throughput is N/M times smaller.

It should be noted that when a recycling converter only has one comparator (one bit) per stage, the difference between that configuration and a (modified) successive approximation converter becomes rather vague.

In addition to truly one-stage recycling converters, which implement N bit conversion in N/M cycles of an M bit stage, hybrid or partially-recycling converters can also be built, like depicted in figure 38. These hybrid converters use a recycling approach, involving more than one stage. these stages form a partial pipeline, of which the last residue is fed back to the first stage. If the total number of bits is N , the number of stages is L and the number of bits per stage is M , the signal will have to be recycled $N/(M L)$ times.

4.7. Time-Interleaved Converters

A recycling converter is related to a pipelined converter as far as the conversion algorithm is concerned, but implements it in time (successive recycling of the residue back to the input) rather than in space (consecutive, physically different stages, operating in parallel). The utilization of a recycling stage is shown as a function of time in figure 39. The utilization of the different stages in a pipelined converter is shown in figure 40. In both cases, it is assumed that a two-phase clocking scheme is used (Φ_1 and Φ_2). Each stage samples the input signal during Φ_1 , processes it during Φ_2 , and has the residue available and stable at its output during the next Φ_1 . Obviously, other clocking schemes could be used in practice, depending on the specific hardware organization.

It is clear that in the recycling approach, space (chip area) is traded for conversion rate (we assume that in both cases, the maximum speed of one stage is identical). In both approaches, the ratio of area to sampling rate (A/f_s) is constant. One could wonder if the trade-off could also be performed the other way, i.e. using a larger area (more stages) in order to obtain faster conversion up

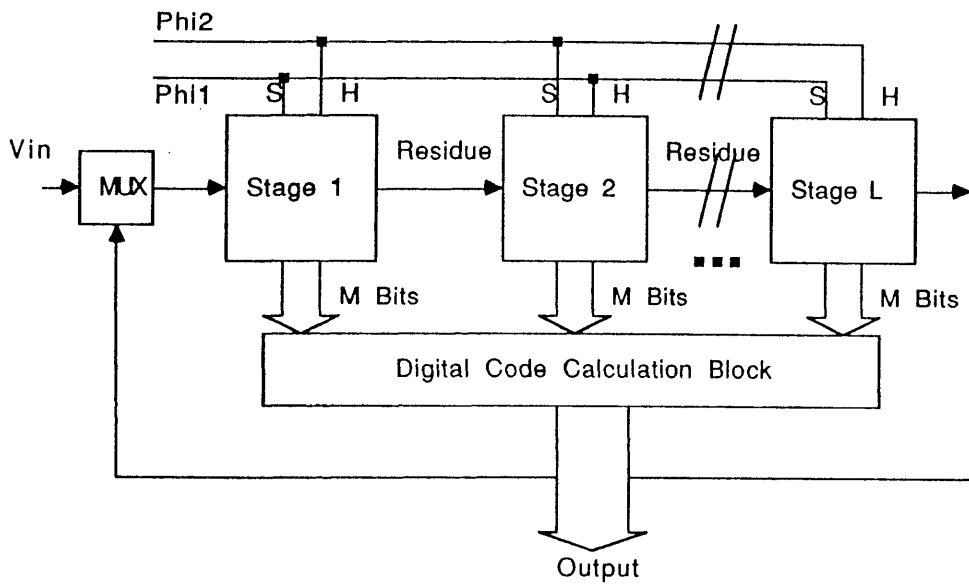


Fig. 38: Partially Recycling (Hybrid) Converter

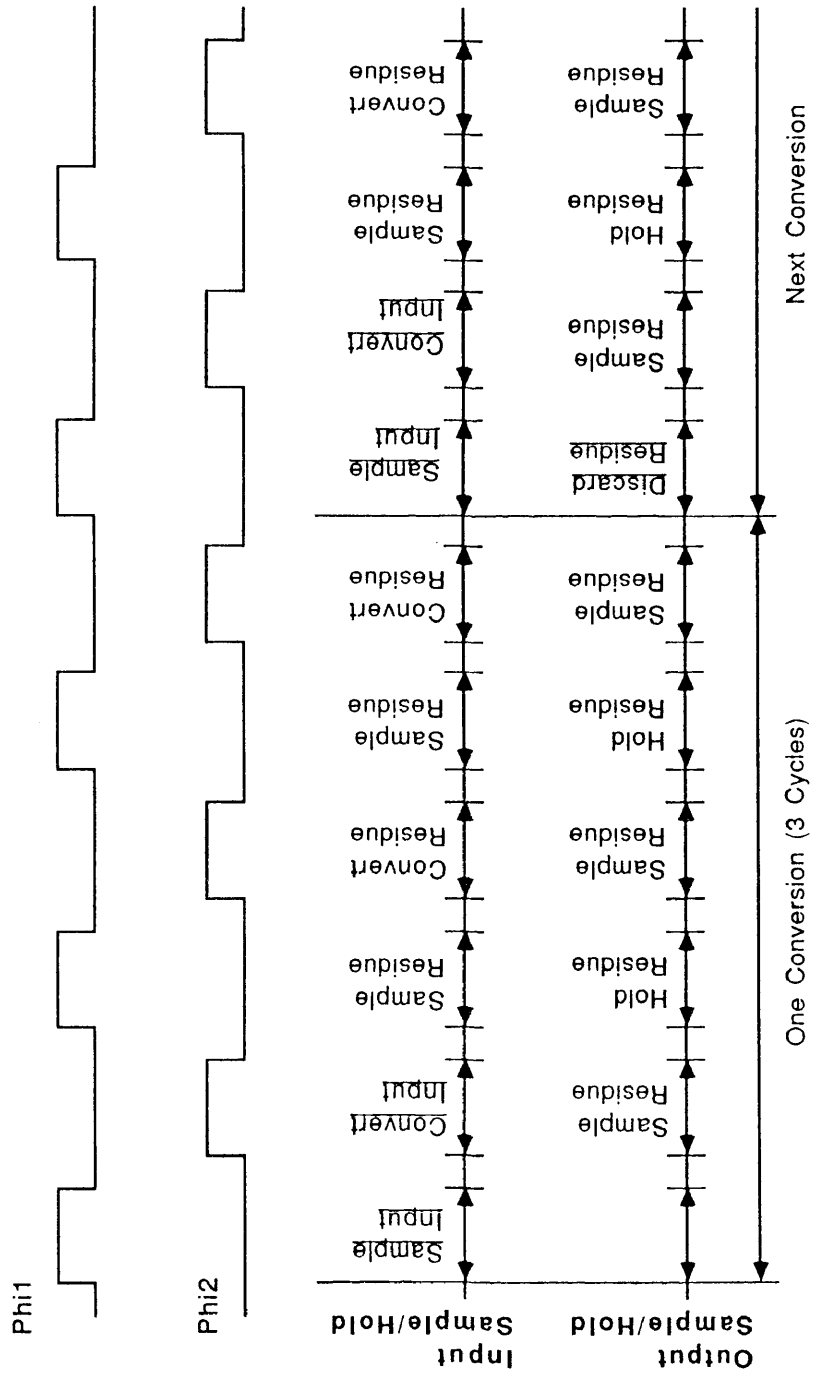


Fig. 39: Timing of a Recycling Converter

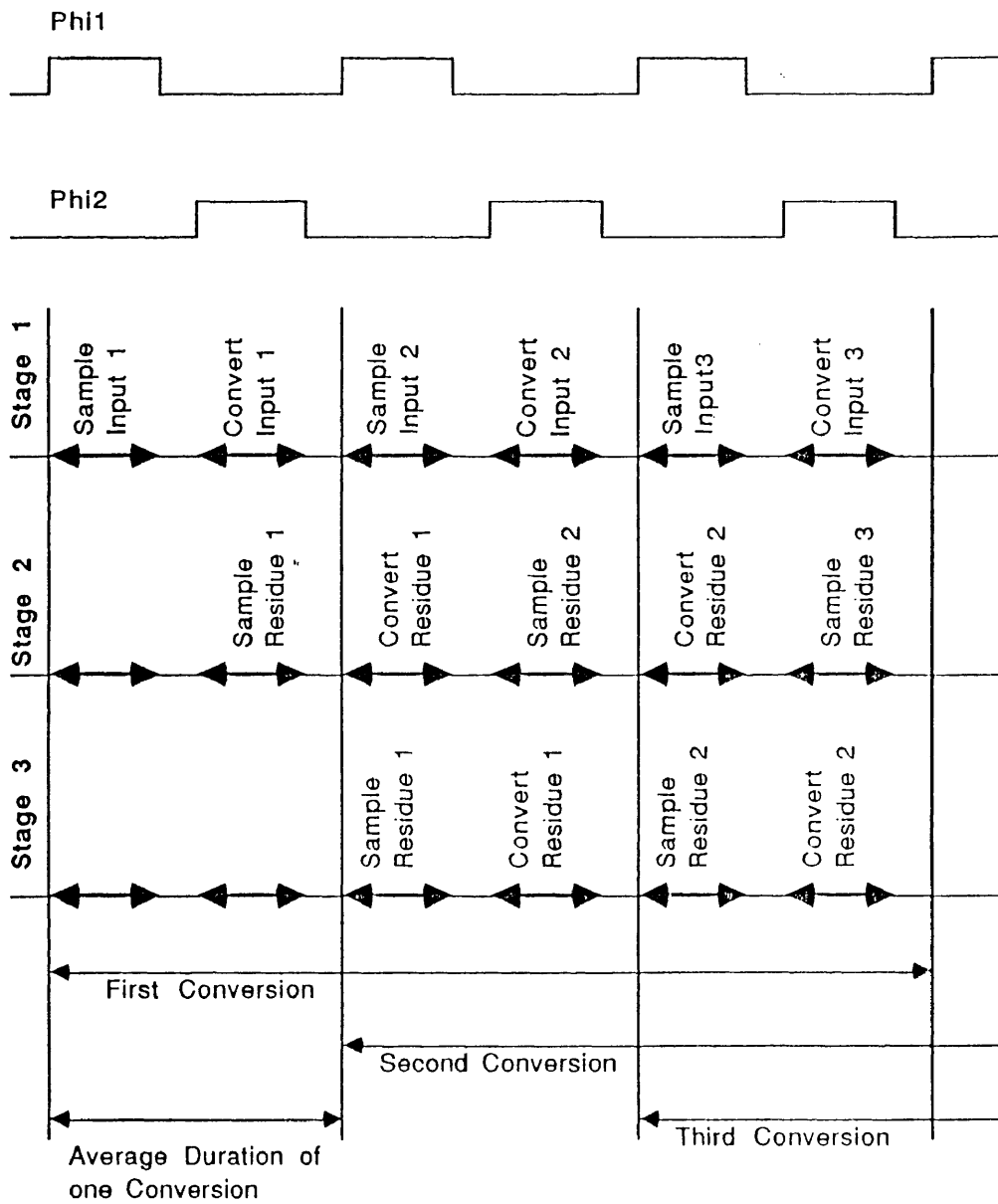


Fig. 40: Timing of a Pipelined Converter

to the same resolution.

A configuration that implements precisely that, is the time-interleaving of several, nominally identical, pipelined converters. The principle is illustrated in figure 41, for an interleaving depth of 4 (4 pipelines in parallel). If one pipeline clock period is equal to T , the maximum throughput (sampling rate), f_s , of each pipeline is $1/T$, since each stage needs a time of T to process its input signal and generate an output (residue).

The first pipeline is clocked so that its first clock phase, Φ_1^1 would start at times $t = k T$, with k integer. (Φ_2^1 starts about one half clock period later, at $t = (k + 1/2) T$.) The second pipeline is clocked so that Φ_1^2 would start at $t = (k + 1/4) T$. Φ_2^3 starts at $t = (k + 2/4) T$ and Φ_1^4 at $t = (k + 3/4) T$. In other words, the clocking of each pipeline is delayed by one fourth clock period with respect to the previous one.

The result is that every $t/4$, one pipeline samples the input signal. If the output bits of the four pipelines are assembled (multiplexed) in a similar way, an ADC with sampling rate $f_s = 4/T$ is obtained, at the price of quadrupling the area. (Also, the logic circuitry needs to be four times faster, but this is usually easier to achieve than making the analog signal path four times faster.)

4.8. Oversampling Converters

From the limited examples discussed above, it becomes apparent that different A/D converter architectures implement various trade-offs between conversion speed (data rate) and circuit complexity (silicon area). In particular, it is clear that recycling converters, single pipelines and time-interleaved converters, built with the same basic components (converter stages) are characterized by a roughly value of their area -conversion time product.

Interestingly enough, this is not the only trade-off. There also exists a direct tradeoff between resolution (number of raw output bits per sample) and conversion speed. This is clearly the case with successive approximation converters and recycling converters. The more approximation steps, or the more cycles are performed, the more bits of resolution the conversion result will have.

Another, although seemingly less efficient, way to trade off conversion speed against resolution is the following. One could use a very fast, low resolution

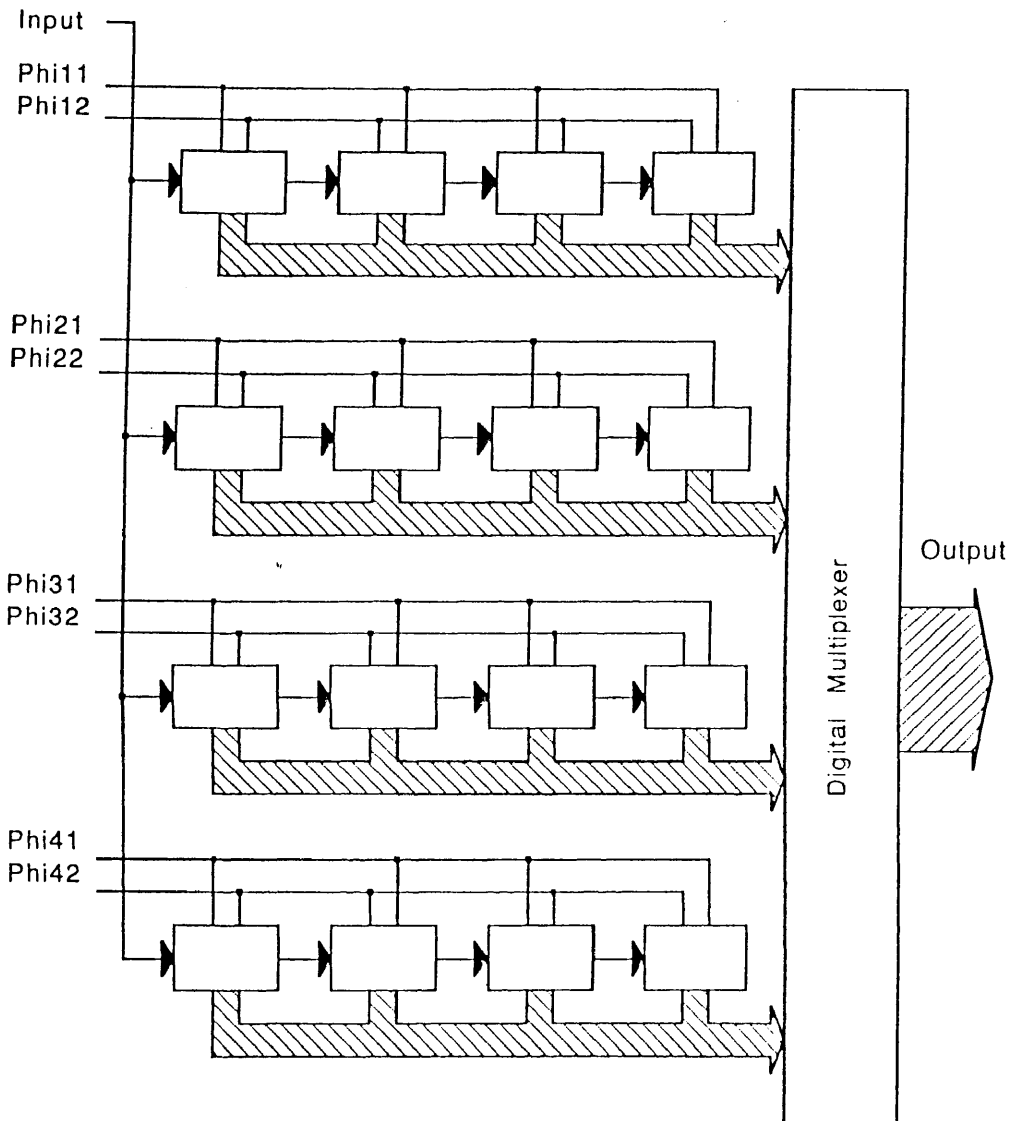


Fig. 41: Time-Interleaved Pipelined Converter

converter to take a large number of samples of an analog signal. The corresponding digital outputs are acquired at actual speed. The process is called oversampling, since more samples are taken of the analog signal than needed to achieve the eventual desired output (digital) data rate.

The digital values can be combined into successive groups, and every time, some kind of average of the values within each group can be computed, using a digital filter. In its simplest form, such filter would be a straight accumulator-adder, which adds up a number of successive sampled digital codes.

The averaging operation can increase the accuracy of the conversion, provided the input signal varies slowly compared to the actual sampling rate. An additional advantage of this method is that some of the noise present in the analog signal can be reduced as well. The process is called decimation, since a large number of (low-resolution) samples is transformed into a smaller number of (high-resolution) samples.

The precise theory behind decimation is strongly related to digital filtering and frequency-domain analysis. This theory, as well as the criterion to determine the minimum required oversampling rate will not be derived here. However, it should be mentioned that oversampling and decimation provide the basis for an entirely different class of A/D converters.

In the extreme case, it is conceivable to build a basic converter with only one bit resolution (one comparator), but oversampling the input signal to such extent that decimation would still provide an accurate conversion result for a certain class of useful input signals.

The disadvantage of the scheme is obviously the required high basic conversion rate, and the need for a sophisticated digital decimation filter. However, the advantage is that the *analog* portion of the converter is simplified to the extreme. Since only one comparator (one bit output) is used, component mismatch and associated non-linearity or distortion are minimized. The fact that a digital filter is required is less of a concern, since the design of such filters is well understood and *not* subject to the parasitic effects that limit the performance of analog circuits.

The most common class of oversampling converters are the sigma-delta converters [49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59]. The basic scheme of such

converter is shown on figure 42. The analog section of the converter consists of an integrator, a comparator and an analog summing element. The comparator is actually followed by a latch (not shown), which freezes its output at periodic time intervals. The term sigma-delta refers to the fact that a discrete-time difference signal is formed (delta operation) and that this difference is integrated over a certain amount of time (sigma operation).

The peculiarity of this set-up is that the comparator output is used as the one-bit conversion result (going into the digital decimation filter), as well as a two-level analog approximation of the integrator output signal. From a conceptual point of view, the system of figure 42 can be represented as in figure 43. The system acts as an analog filter with feed-back, of which the transfer function is

$$V_2(s) = V_1(s) \frac{1}{\frac{s}{A} + 1} \quad (59)$$

This clearly represents a low-pass filter, described here in the continuous-time domain (s). In actuality, due to the sampling operation of the comparator output, this system would be modeled more accurately in the discrete-time domain (z). For this simplified discussion however, the distinction is irrelevant.

The one-bit quantizer can be thought of as an element that injects quantization noise into the system (figure 44). For non-periodic input signals, the frequency spectrum of this quantization noise is fairly uniform, and the noise can adequately be modeled as wide-band white noise, added to the integrator output signal. One can easily verify that the transfer function between the summing node of the noise and the output is

$$V_2(s) = V_n(s) \frac{s}{s - A} \quad (60)$$

This is the representation of a high-pass filter. As a result of the difference response of the system to the input signal and the equivalent noise signal, the two effects can be separated. Through judicious choice of the integrator gain and the sampling rate (higher sampling rates spread out the same amount of noise over a larger frequency band), a system can be built in which the quantization noise only becomes significant for frequencies far above the maximum input signal frequency.

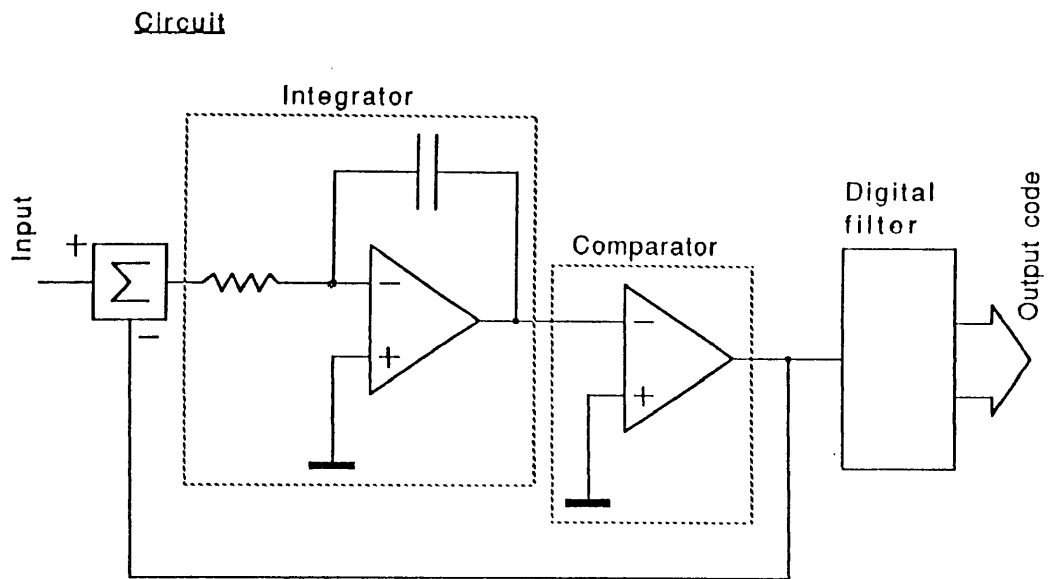


Fig. 42: Basic Sigma-Delta Converter

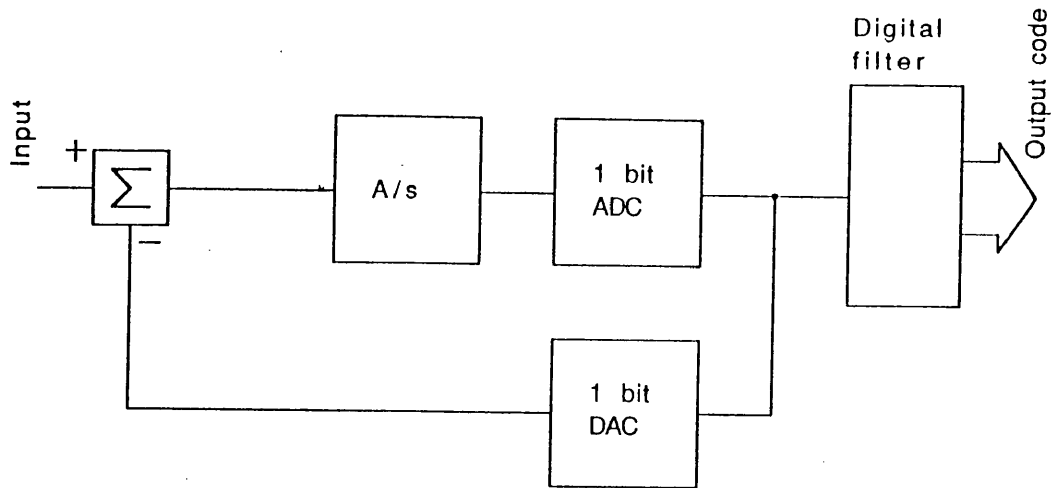


Fig. 43: Equivalent Sigma-Delta System

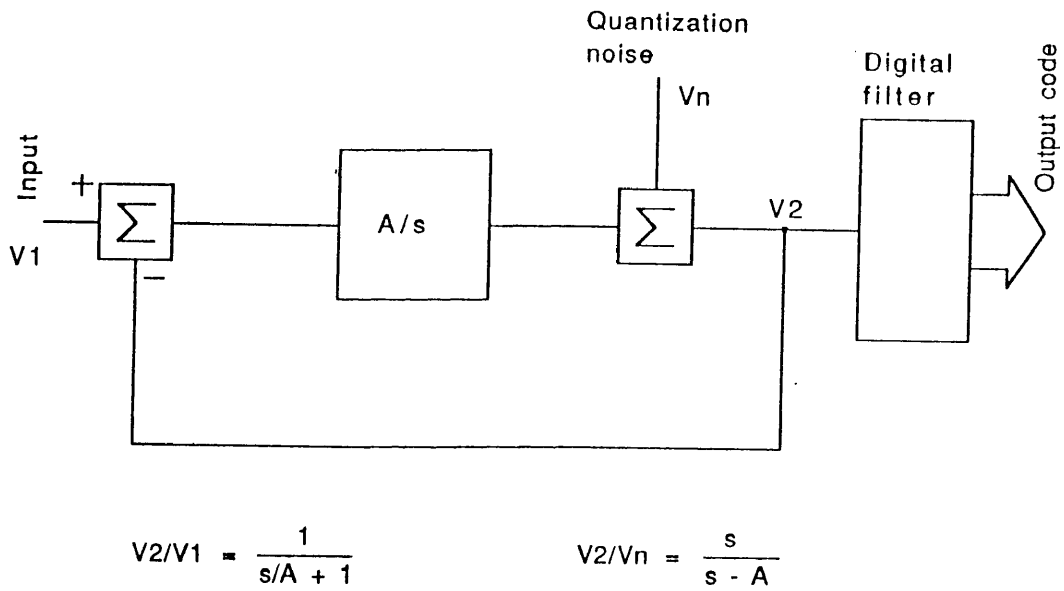


Fig. 44: Model with Quantization Noise

Low-pass filtering and decimation [49] in the digital domain can then be used to reject most of the quantization noise, while retaining an accurate digital representation of the input signal. In its most elementary form, the decimation filter could be a simple binary up/down counter. The up/down control line is connected to the comparator output, and the counter is reset every M samples, with M the desired oversampling ratio. Obviously, in practice more sophisticated, higher performance schemes are used.

A lot of research has been done on sigma-delta converters over the years, and numerous improvements have been made to the basic scheme. Most practical systems now use more complex analog structures, with more than one integrator [50, 60, 61, 62] and multiple feed-back loops. As opposed to the basic structure of figure 42, the stability of multiple-loop structures is of major concern. However, the quantization noise rejection can be dramatically improved. In some cases, performance can be enhanced even further using multiple-bit quantizers instead of a single comparator.

As mentioned earlier, the main advantage of sigma-delta converters (and oversampling converters in general) is the simplicity of the analog components. Integrators, summers and comparators can be built very reliably. In addition, gain or offset errors in any of these blocks will not influence the overall linearity of the converters scheme. It will be shown in chapter V that these are precisely the non-idealities which have traditionally limited the performance of high-speed, multi-step or pipelined structures.

Sigma-delta converters are usually operated at oversampling rates (comparator clocking rate divided by effective digital filter output rate) of several tens to several hundreds. Due to the high comparator sampling rates required (sometimes several MHz, without adding any distortion to the signal), the effective data output rate has been limited to the 10 kHz-100 kHz range. However, integral non-linearity values far beyond the 16 bit level have been reported. As a result, oversampling converters have been a favorite architecture for digital high-fidelity audio applications, in particular digital recording and compact disc systems.

Although very active research is being done on reducing the oversampling rate and increasing the overall conversion rate of sigma delta converters, it seems clear that without any revolutionary break-through, their performance will limit them

to high-accuracy, low-frequency applications. On the other hand, new research described in this dissertation seems to indicate that the accuracy of inherently fast pipelined architectures can be improved dramatically (by orders of magnitudes) through the use of new digital error characterization and correction techniques. It is expected that these new techniques could dramatically shift the application field of data converters from sigma-delta to pipelined architectures.

4.9. Conclusion

In this chapter, several common analog to digital converter architectures were introduced in a systematic way. The discussion started with the flash converter, which was later extended to multi-step converters, of which pipelined converters are one example. The successive approximation converter was discussed, and the relationship with multi-step architectures demonstrated. The recycling converter was introduced as a special case of the pipelined converter, with only one stage. Finally, the time-interleaved converter, consisting of several pipelines operating in parallel, was discussed. It was shown how the pipelined, recycling and time-interleaved converters can each be built from identical stages, with different trade-offs between area and conversion speed. Finally, oversampling converters were introduced from a general point of view. Sigma-delta converters were briefly discussed as a particular implementation. It was demonstrated how new error correction techniques for pipelined converters could eventually make these more attractive than the conventional sigma-delta architectures.

CHAPTER V

ERROR MODELING IN MULTI-STEP CONVERTERS

5.1. Introduction

The previous chapter introduced several important analog to digital converter architectures. One of them was the multi-step converter (of which most pipelined converters are special cases). The multi-step architecture is very important, for its ability to implement compact, high-resolution converters. The second part of this dissertation will concentrate on one particular implementation: high-speed, high-accuracy pipelined converters. In order to obtain such high accuracy, a very good understanding of possible error mechanisms in the analog data path is needed [8]. This chapter describes these error mechanisms, and provides practical methods to evaluate the errors in actual analog to digital converters. Subsequent chapters will focus on how to correct (compensate) the effect of errors.

5.2. Origin of the Errors

When the static transfer curve of a multi-step converter (e.g. a pipelined converter) is measured using a low-frequency histogram test, variations from the ideal pattern are almost always seen. In particular INL errors (variation of the actual code transition levels with respect to the ideal, or designed-for transition levels) are not exactly zero, like they should be for an ideal converter. Instead, the INL values exhibit particular patterns when plotted as a function of the bin number.

These imperfections can be traced to incorrect values in the flash and DAC subsections of particular stages in the converter, or to incorrect gains in the amplifiers between two stages. It should be

noted that in a pipelined converter, the interstage amplifiers perform a sample/hold operation as well, but this is not essential. Non-pipelined multi-step converters have successfully been used in practice. In some cases (sub-ranging converters), there is no interstage amplifier. However, these cases still fit the same model, with interstage gain values of 1.

Offsets in the interstage amplifiers can also introduce non-linearity errors, but the effect is comparable to that of a DAC error and should not be treated separately. This can easily be visualized as follows. If the interstage amplifier has an input-referred offset x_o , the situation is equivalent to having an ideal amplifier, and a value x_o being subtracted from the input signal. The result is the same as if the DAC levels (subtracted from the input signal to calculate the residue) had been increased by x_o . As a result, our analysis of static errors will be limited to flash, DAC and gain errors.

Each stage of a multi-stage ADC takes an input signal, converts it to an M bit code (after conversion from thermometer coding to binary) and passes the residue of the conversion on to the next stage. Figure 45 shows the basic components: flash section (reference string, comparators and latches), DAC section and interstage amplifier. The amplifier is shown at the output rather than the input, for easier modeling of errors later on. Since these amplifiers are actually located at the interface between two stages, there is no significant difference between the two conventions.

The process of residue calculation as a function of the input signal to a stage, can be represented on a graph (figure 46). The X axis represents the input signal. The code transition levels of the stage are displayed in *lsb* along this axis, using the conventions of chapter I. It is clear that since the stage contains an elementary flash converter, all definitions introduced in connection with ADC transfer functions can be maintained to describe one stage. To each bin of the flash converter corresponds an output code, which is converted to a voltage (or current) by the DAC section. We will assume that the code for bin -1 (under-range condition) is 0 and for bin 2^N (over-range) is $2^N - 1$, which is typical of flash ADC's without error correction.

The DAC output is displayed in the direction of the Y axis, as a horizontal line segment above the corresponding ADC bin. The same scale (*lsb*) is used for the Y axis as for the X axis. The residue of the flash/DAC combination can be found as the difference between the $y = x$ line and the horizontal line segments. This can easily be verified by the fact that the residue (Y direction) is equal to the input signal (X direction), minus the appropriate DAC level, which is represented by a line segment above the relevant bin. The residue function is shown on figure

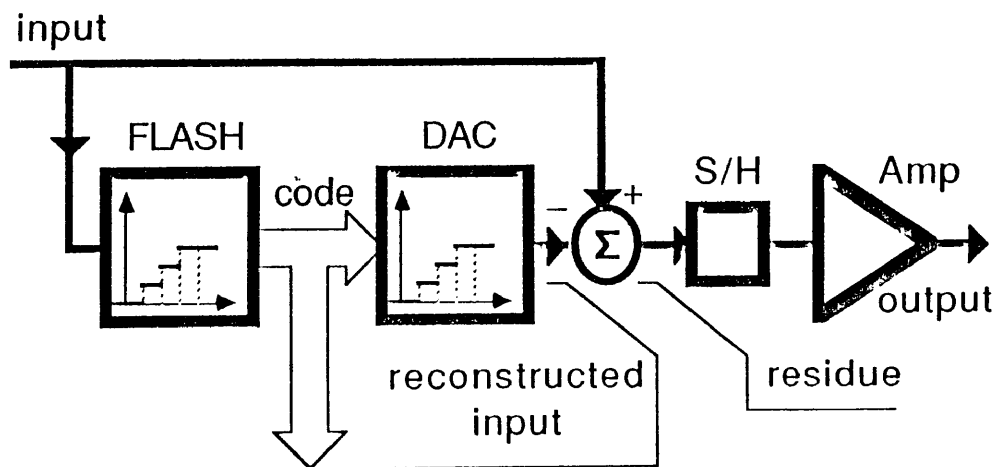


Fig. 45: Basic Multi-Stage Scheme

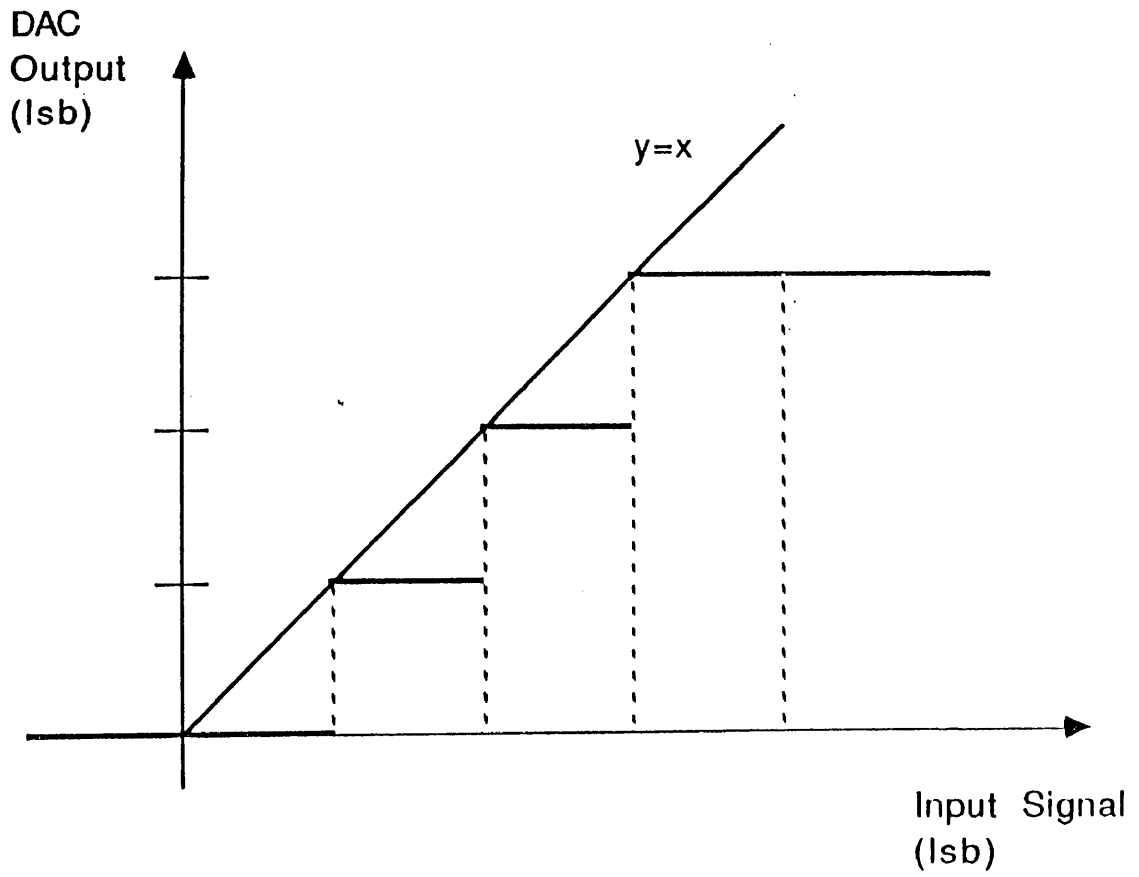


Fig. 46: Residue Calculation

47, for an ideal 2 bit stage. Figure 48 shows the same residue, after multiplication by 4 (2^N), which represents the effect of the output amplifier.

5.3. Error Analysis Methodology

The following analysis of error mechanisms will use this relationship between input signal and residue. It will investigate the effect of incorrect flash levels, DAC levels and gains upon the shape of the residue function. Since the residue is passed on to subsequent stages, the errors will be propagated throughout the converter.

This propagation can easily be related to the histogram concept, which is one of the reasons why the histogram theory was introduced before any particular ADC architectures were discussed. We will assume that a histogram test is performed by applying an input signal of constant magnitude distribution to the input of the first ADC stage. If this stage were ideal, the samples would be evenly spread over its M bins, and the histogram of the output codes would be flat.

In practice, the histogram test would be performed using the full output codes of the converter, including the bits generated by subsequent stages, as well as the bits from the current stage. As a consequence, two things will affect the overall histogram:

- The transition levels of the stage in question, being the effective comparator trippoints.
- The magnitude distribution (probability density) of the output signal of that stage (the residue), since this is the signal that will be passed on to others stages.

One could do an error analysis relying on the precise number of bits of the total converter (N), by considering the effect of errors in the first stage on the effective bin widths of each of the 2^N bins of the converter. However, this would limit the discussion to specific configurations, rather than providing a general description. Instead, we will investigate what would happen in a limit case, if the first stage were followed by an infinite number of subsequent stages, each of them being infinitely accurate (no flash, DAC or gain errors).

Since the total number of bits of such hypothetical device is infinite, the bin

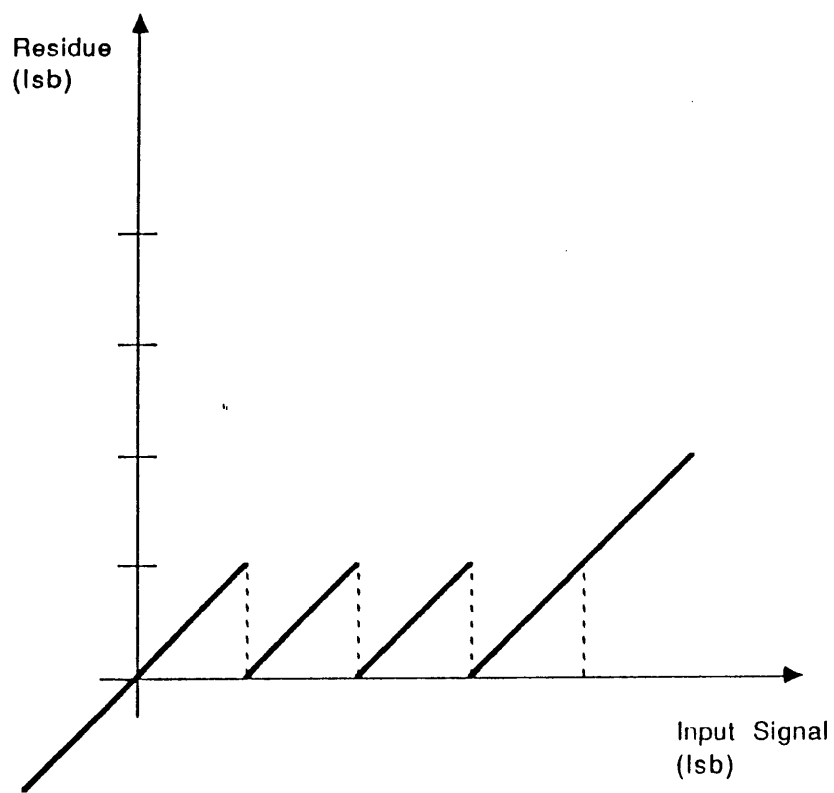


Fig. 47: Residue Function of Ideal Stage

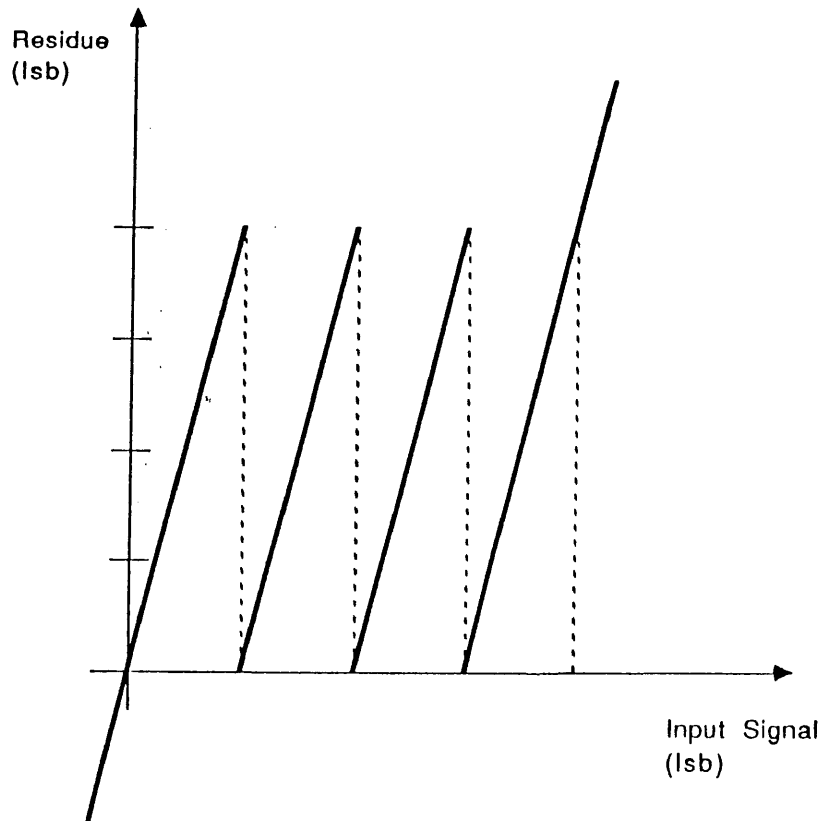


Fig. 48: Residue, after Gain Stage

widths would be infinitely small, and the expected number of samples ending up in each bin during a histogram test, would be zero. In order to overcome this restriction, we will consider the normalized bin widths (nominally equal to 1) of the device. Since there are an infinite number of bins, the normalized bin widths will form a *continuous* function of the signal magnitude, rather than a discrete one.

It will be shown that, just like histogram tallies, these normalized bin widths are proportional to the probability density of the signal applied to the second (ideal) part of the converter. As a result, the bin widths will be determined by the probability density function of the residue from the first stage. Since the first stage divides the nominal input range of the signal into M bins, each corresponding to a different set of most significant bits, the probability density function of the residue will have to be determined separately for each bin of the first stage. The probability density will be scaled in such a way as to be exactly equal to 1 for an ideal first stage, in order to reflect the nominal, ideal bin widths of the whole converter.

Figure 49 shows the normalized probability density functions (*pdf*'s) of the residue for each bin of an ideal first stage (2 bits). The integrals of the pdf's over their respective bins are 1, since the width of each bin is exactly 1 *lsb*, and the pdf's were normalized to 1. The concatenation of pdf curves can be seen as the limit to which a normalized and perfectly accurate (∞ points) histogram would converge if the ADC stage were followed by other stages, with infinite resolution. For this reason, we will sometimes refer to the concatenation of pdf's as the "limit histogram" of the stage.

Figure 50 shows the limit histogram curve after subtraction of 1. It is clear that since the normalized histogram represents bin widths, the result of the subtraction represents the limit DNL curve that would be obtained from an infinitely accurate histogram test with infinite resolution. Ideally, this limit is 0. Figure 51 shows the integral of the DNL curve, or the limit INL curve. Ideally, this curve is also equal to 0. It will be seen how errors in the first stage of the converter will affect limit histogram, limit DNL and limit INL curves in a very characteristic way.

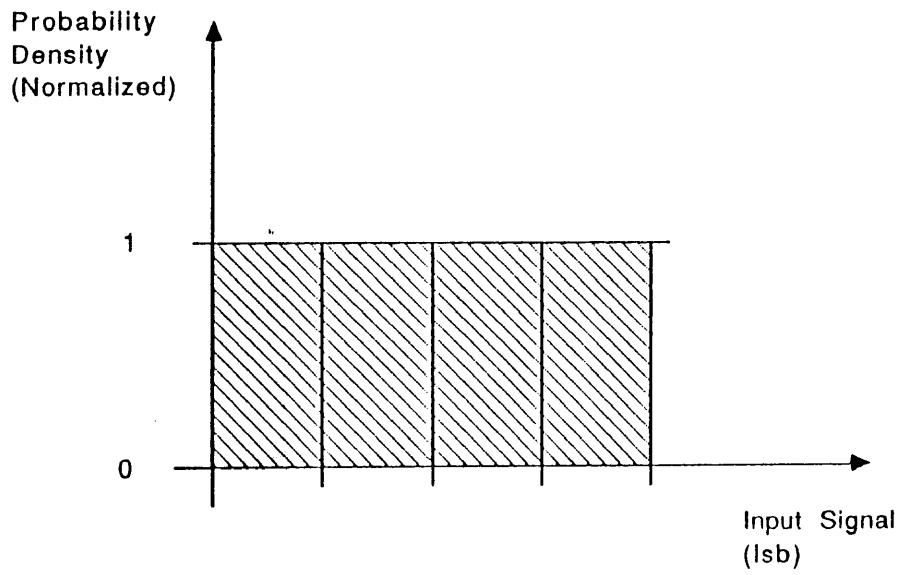


Fig. 49: Probability Density Functions

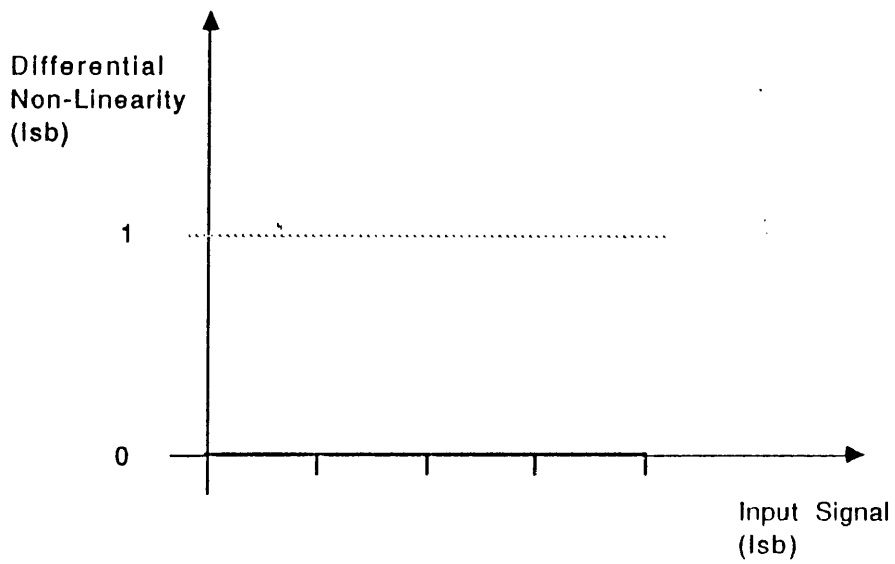


Fig. 50: Limit DNL Curve

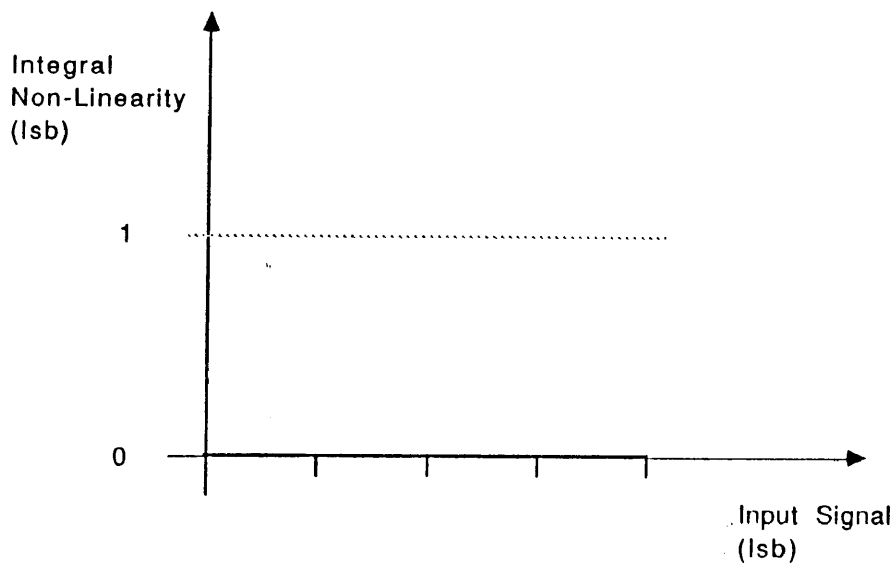


Fig. 51: Limit INL Curve

5.4. DAC Errors

Figure 52 shows the same considerations for a stage in which one of the DAC output levels larger than ideal. In this case, there is a positive DAC error when the input signal is in bin 1. The magnitude of the error is subtracted from the ideal value of the residue for that bin.

Since at this point, we are looking at the effect of errors in the first stage only, we will assume that the stage under consideration is followed by other stages with limited input signal range. In other words: the input range of the next, ideal stage is exactly equal to the *nominal* range of the residue from the first stage. Any signal outside of that range will generate a code (least significant bits) equal to the code of the value at the range boundary, either $00 \cdots 0$ or $11 \cdots 1$. This can be represented as if the residue were "clipped", in the sense that any value outside of the nominal range will have the same effect as the minimum or maximum value of the range. We assume that subsequent stages do not recognize over-ranging or under-ranging conditions.

One can see how the pdf within the corresponding bin is redistributed, by looking at the residue function. For an input signal within bin 1, a small extra value equal to the DAC error is subtracted from the residue. As a result, the residue will never reach the upper end of the range while the input signal is in that bin. There is a "gap" in the normalized pdf at the higher end of the bin. One can verify that the area (integral) of this gap is equal to the magnitude of the DAC error. If one actually were to perform a histogram test using a converter with the given stage as first stage and a finite number of subsequent stages, this gap in the pdf (limit histogram) would actually show up as a number of "missing codes" or codes that never occur.

At the lower end of the bin, the pdf will have a "peak" (delta function) with area equal to the DAC error. This is due to the fact that residue values below the lower range boundary are equivalent to the value at that boundary, of which the probability density becomes infinite. In an actual histogram, this peak would stretch over exactly one bin, and show up as a disproportionate sample count. One can verify that if the DAC error had been negative, gap and peak would have been interchanged.

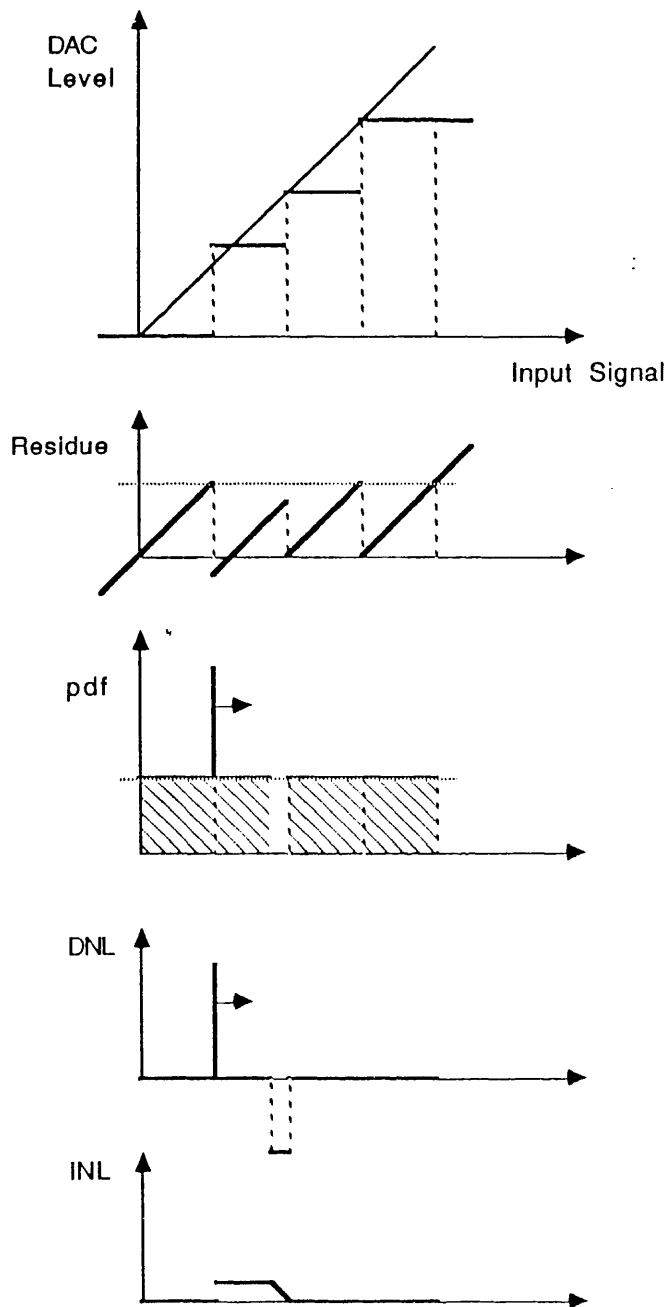


Fig. 52: Stage with a DAC Error

The effect of the DAC error is similar on the limit DNL curve ("peak" and "gap"). The effect on the limit INL curve can be found by integrating the limit DNL curve. One can see how it mainly creates a shift in INL level in the middle of the bin (with a small, linear transition at the end). The amount by which the INL function is raised, is equal to the value of the DAC error.

5.5. ADC Errors

Figure 53 shows the effect of an error in the flash section. In this example, there is a negative INL error on code transition level 1, meaning that the actual level is lower than nominal. As can be seen on the residue plot, the residue will not reach its maximum value while the signal is in bin 0, since the bin itself has shrunk. The residue pdf has a gap at the end of the bin, representing missing codes due to missing residue values associated with bin 0. At the same time, bin 1 has grown, and it now contains residue values below the nominal range. This will create a delta peak in the normalized pdf, at the lower end of bin 1. One can verify that the area of gap and peak is equal to the value of the error on the comparator level. The value of the normalized pdf (limit histogram) in the middle of bins 0 and 1 is unaffected, just like in the case of a DAC error.

In general, a negative INL error on ctl_i causes a gap at the end of bin $i-1$ and a delta peak at the beginning of bin i , both with an area equal to the magnitude of the error. A positive error on ctl_i would cause a peak at the end of bin $i-1$ and a gap in the beginning of bin i . The same peak and gap show up in the limit DNL curve, while the limit INL curve exhibits a small, positive or negative triangular peak, pointing in positive direction for a positive error and in negative direction for a negative error.

5.6. Gain Errors

The term gain error is used to designate an error in the gain factor by which the residue is amplified between ADC stages. Figure 54 shows the influence of a gain error greater than 1, meaning that the residue is overamplified. A gain error affects all bins in the same way.

As a general rule the constant level of the pdf within a bin is divided by the extra gain factor ϵ_A . This is due to the fact that the slope of the residue,

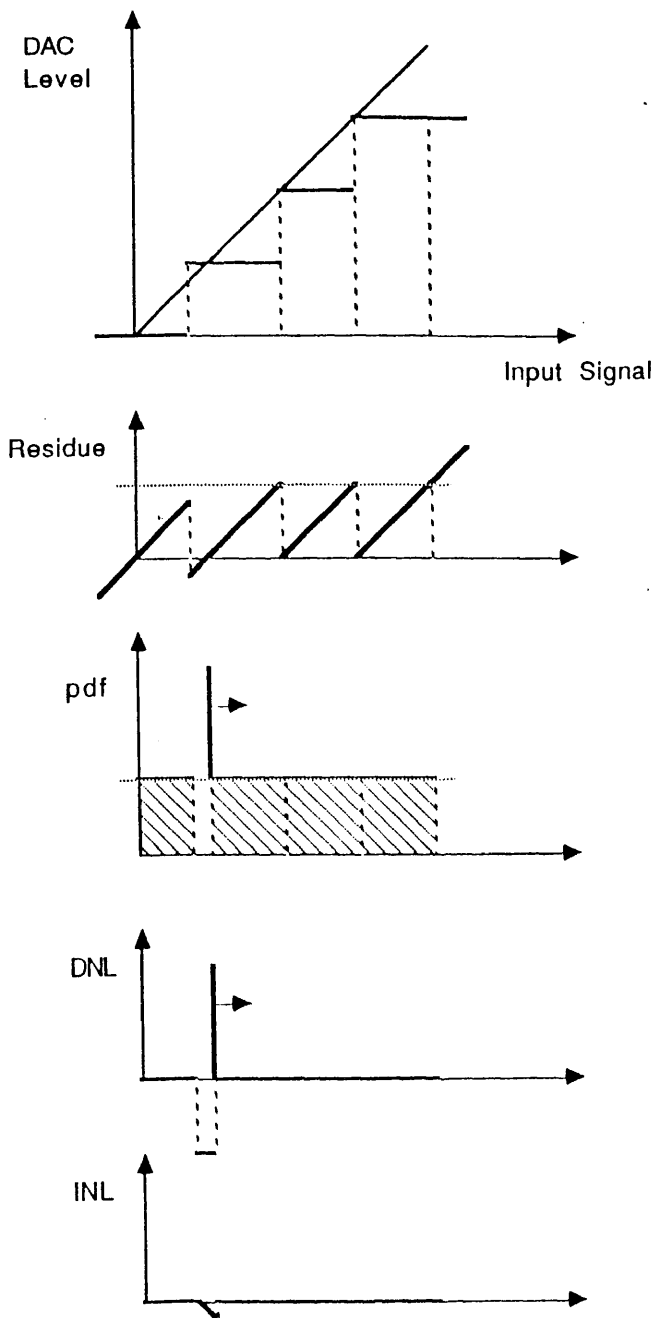


Fig. 53: Stage with a Flash Error

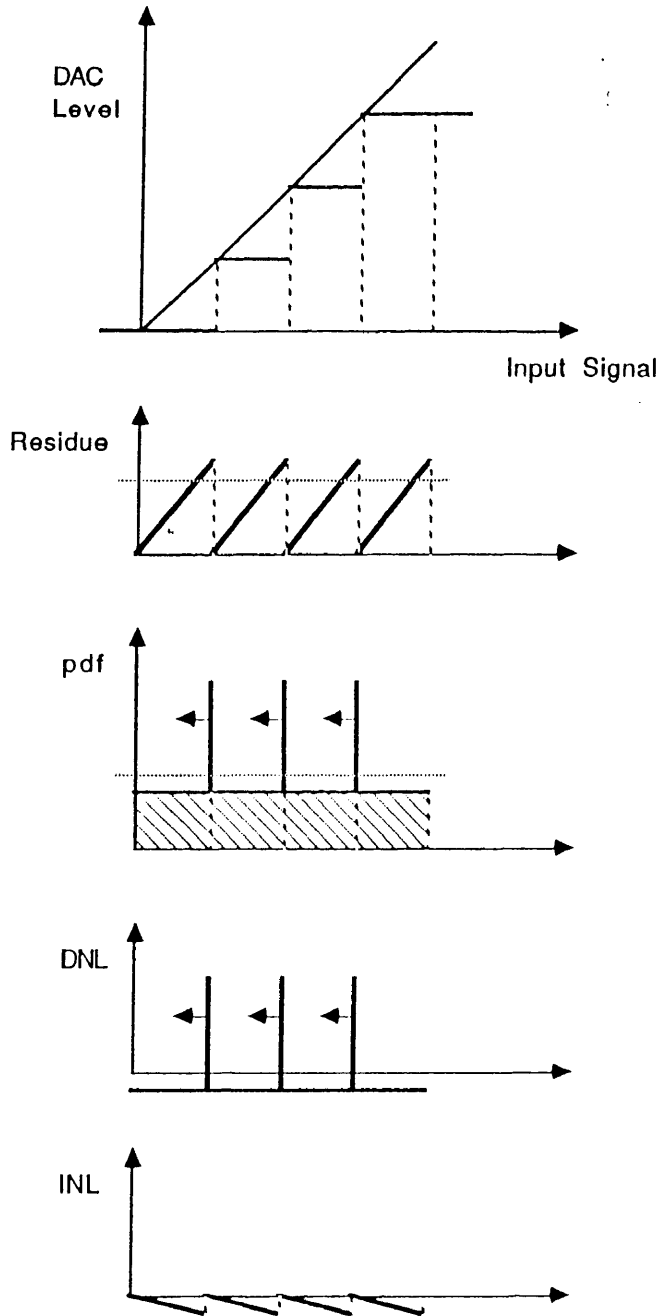


Fig. 54: Stage with Excessive Gain

as a function of the input signal, is increased by a factor ϵ_A . This results in a corresponding decrease in the pdf of the residue. The reason is the following. We assume that the pdf of the input signal is constant. The residue represents a function (transformation) of the input signal. It is well-known that in such case, the pdf of the transformed signal is inversely proportional to the derivative of the transforming function.

If ϵ_A is greater than 1, a delta peak with area $1 - \frac{1}{\epsilon_A}$ will appear at the higher end of each bin. This is due to the fact that the residue exceeds its nominal range at the end of the bin. If we assume that this results in the effective clipping of the residue, the pdf will become infinite in those points. The area of the peak is determined by the fact that the integral of the pdf over each bin must be 1 (a gain error does not affect the width of the bin, since the code transition levels are not changed). Since the pdf level within the bin changes from 1 to $1/\epsilon_A$ (resulting in an area of $1/\epsilon_A$ as well), the area of the peak must be $1 - 1/\epsilon_A$.

If ϵ_A were smaller than 1 (interstage gain smaller than nominal), the pdf within the bin would be increased by ϵ_A , and a gap of area $\frac{1}{\epsilon_A} - 1$ would appear at the higher end of each bin. One can easily verify on figure 54 and 55 that a gain error will cause the limit INL curve to look like a sequence of positive or negative triangular shapes, with discontinuities at the transitions between the major bins.

5.7. Combined Effect of Errors

In order to find out how individual errors combine, one could express the pdf of the residue (the limit histogram) of each bin of an ADC stage in a condensed way. Within a bin, the pdf can show either a peak or a gap at the lower end of the bin, followed by a section of constant pdf and finally a peak or a gap at the higher end of the bin. (Note that it is crucial to distinguish between peaks at the higher end of a bin and peaks at the lower end of the next bin!) This information could be organized into a 3-dimensional vector.

The first dimension will be referred to as the lower *anomaly* of the pdf in that bin, being 0 if there is no peak or gap at the lower end of the bin, positive if there is a peak (with magnitude equal to the area of the peak), and negative if there is a gap (with magnitude equal to gap area). The second dimension will be the mid-bin pdf level, and the third dimension the higher *anomaly* (equal to peak or

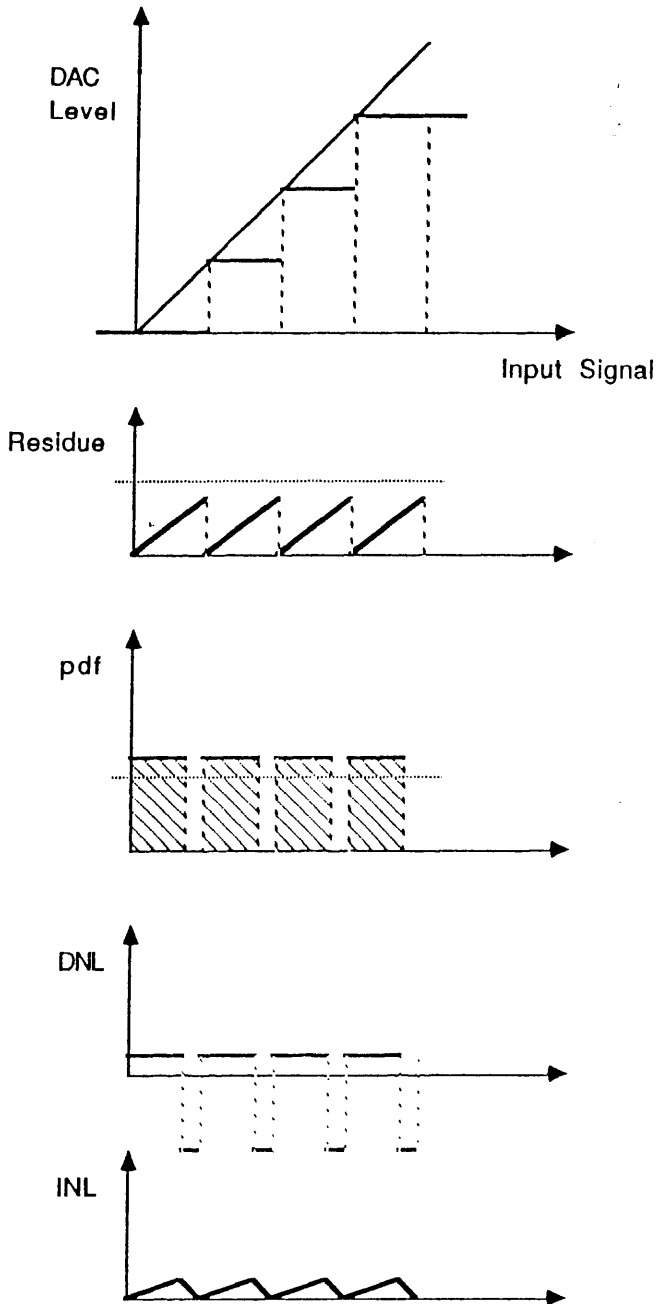


Fig. 55: Stage with Insufficient Gain

gap area like for the lower anomaly).

The 2^M (one for each bin) such pdf vectors will be referred to as v_i . For an ideal, errorless stage, $v_i = (0, 1, 0)$ for $0 \leq i < 2^N$. An isolated DAC, ADC or gain error affects one or more of the v_i . The descriptions given above can be translated into vector notation.

An isolated DAC error of ϵ associated with bin i is characterized by

$$v_i = (\epsilon, 1, -\epsilon) \quad (61)$$

An ADC (flash) error of ϵ on ctl_i by

$$v_{i-1} = (0, 1, \epsilon) \quad , \quad v_i = (-\epsilon, 1, 0) \quad (62)$$

And a gain error of ϵ_A by

$$v_i = \left(0, \frac{1}{\epsilon_A}, 1 - \frac{1}{\epsilon_A}\right) \quad , \quad \text{for all } i \quad (63)$$

This notation can be used to determine how different error mechanisms affecting more than one bin, influence each other. As an example, figure 56 shows how a DAC error (in bin 1) and an ADC error (on ctl_1) influence each other. The pdf peak at the lower end of bin 1 now has an area equal to the sum of the peak areas caused by the individual errors, and there are two gaps (higher end bin 0 and higher end bin 1). This observation is generalized in the following theorem.

5.8. Error Superposition Theorem

The theorem consists of two parts. The first one is a rule for combining the effect of different error types upon one single bin. The second part is a rule for combining the effects of errors affecting adjacent bins, into the total pdf (limit histogram) function for the stage.

1. The pdf vectors for each bin of an ADC stage subject to multiple errors, are found by combining the pdf vectors reflecting the effect of each single error upon that bin. As such, the combined pdf vector of bin i is affected by a DAC error on DAC level i , ADC errors on code transition levels i and $i + 1$, and a gain error on the output amplifier of the stage. The rule for combining the pdf vectors determined by these four potential errors, is the following:

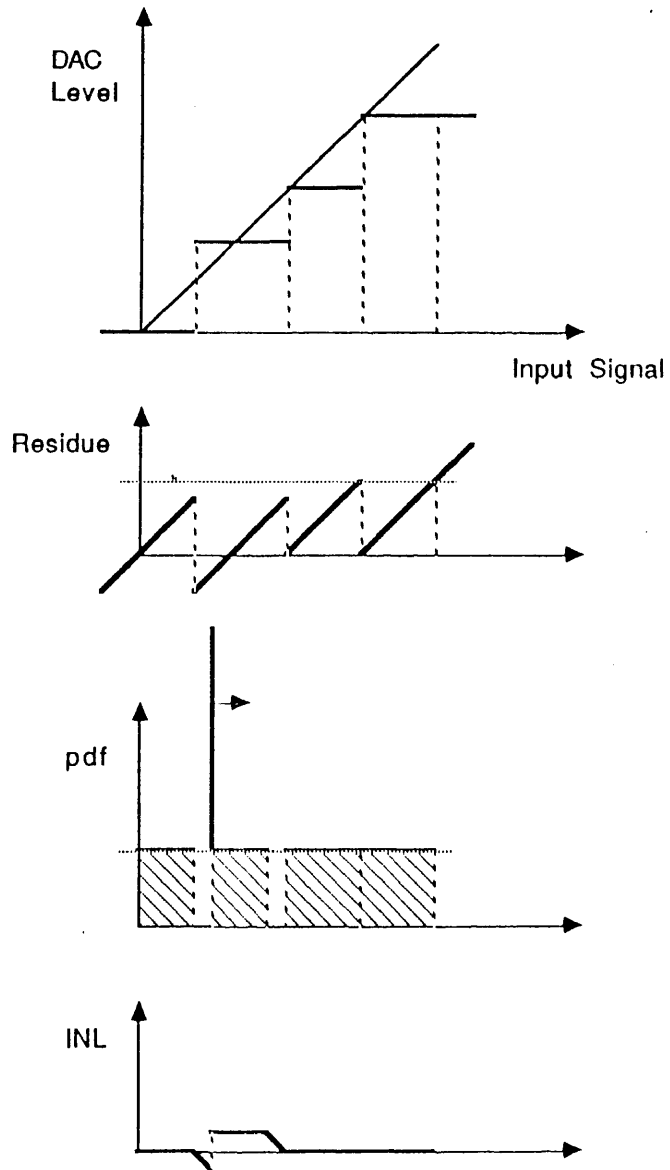


Fig. 56: Stage with DAC and Flash Error

- * lower and higher anomalies are added bin per bin
- * constant parts are multiplied bin per bin

Obviously, this combination rule is associative and commutative, which means that the order in which individual errors are considered is irrelevant.

2. The limit histogram for the stage under consideration is obtained by concatenating the 2^M pdf functions (one for each bin), which in turn are obtained by expanding the pdf vectors, determined according to the procedure outlined in point 1.

The full proof of this theorem will not be given here, since it is at the same time fairly trivial and lengthy. The rule for combining pdf vectors corresponding to the effect of individual errors on each bin can be demonstrated by analyzing the effect of several errors affecting one bin, from a probability density point of view. By translating the effect into vector notation each time, point 1 can be proven. Point 2 results from the property that the limit histogram is proportional to the residue pdf function of each bin.

5.9. Derivation of Per-Stage Errors

We have seen how knowledge of the individual error mechanisms of one stage makes it possible to determine the limit histogram of that stage. The limit histogram is the normalized histogram that would be obtained if the stage were followed by other stages with an infinite number of bits (infinite resolution) and the number of samples taken were also infinite. The normalized histogram reflects the normalized bin widths (in lsb) of the hypothetical converter obtained by adding an infinite number of ideal stages to the (non-ideal) first stage.

Conversely, it will be shown that knowledge of the limit histogram of the first stage, makes it possible to precisely determine *all* DAC and flash errors affecting that stage, as well as the gain error affecting the interstage amplifier at the output.

The gain error is the easiest one to isolate, since the constant pdf levels within any one of the 2^M bins are equal to $1/\epsilon_A$. As a result, inversion of the second value in any one of the pdf vectors of a stage, directly yields ϵ_A . If the vectors are determined experimentally, the second values of all vectors may not be exactly equal, due to measurement or estimation errors (histogram test). In that case,

the M values could first be averaged together to improve overall accuracy.

It is also possible that ϵ_A values for each bin exhibit a deterministic pattern, rather than random variations. This could be due to the fact that the interstage amplifier after the first stage exhibits a certain amount of non-linearity (signal-dependent gain variation). It could also be due to the fact that a histogram test was performed using a non-ideal input waveform (e.g. a ramp with slightly flattened tops, resulting in U-shaped distortion of the histogram). In that case, the different ϵ_A values will provide valuable information about the non-linearity or the distortion, and they should not be averaged together.

Once the gain error has been determined, the other two kinds of errors can be derived by analyzing lower and higher anomalies (first and third values) of the different pdf vectors. In order to mask out the effect of the gain error, a value of $1 - \epsilon_A$ should first be subtracted from the higher anomaly of each vector. This could be done using an average ϵ_A value, or individual values if gain non-linearity or systematic distortion is suspected. Ideally, after this operation, the sum of all lower and higher anomalies should be zero. This can easily be verified by the fact that individual DAC and ADC errors always result in a positive and a negative anomaly value of equal magnitude, resulting in a sum of 0.

However, due to measurement errors during the histogram test, this may not be exactly the case. In order to avoid biased results, a value equal to the sum of all anomalies, divided by $2M$ (twice the number of bins) could be subtracted from each anomaly value. At the same time, the magnitude of this correction term is indicative of the accuracy of the histogram test.

After these corrections, individual DAC and ADC errors can be determined simultaneously for each bin, starting from one end of the histogram. The lower anomaly of bin 0 is normally always equal to 0. This is due to the fact that the first and last bin of a histogram test are always discarded and replaced by an ideal value (implicit normalization of the input signal range). As a result, there cannot be a peak in the very first bin. There cannot be gap either, because if there were, this would have been interpreted as if the input waveform used during the histogram test did not cover the whole input range, and the whole test would have been repeated using a waveform of slightly higher magnitude. The same thing could be said about the higher anomaly of the last bin.

As a result, bin 0 and bin $2^M - 1$ are not subject to DAC errors. Code transition levels 0 and 2^M are not subject to ADC errors. This leaves $2 \cdot 2^M - 3$ independent unknowns to be determined: $2^M - 2$ DAC errors and $2^M - 1$ ADC errors. These unknowns can be calculated from the equations describing the theoretical lower and higher anomalies of each bin as a function of ADC and DAC errors. There are $2 \cdot 2^M$ such relationships (one lower and one higher anomaly for each bin). However, two of them are trivial (lower anomaly of the first bin and higher anomaly of the last bin). In addition, the equations are not independent. They are connected through one relationship, since it is known that the sum of all anomalies is zero. As a result, the $2 \cdot 2^M - 3$ errors could be calculated using a set of simultaneous equations. However, the equations can be simplified considerably.

The DAC error of bin i can be calculated by summing together all the anomalies of the pdf vectors corresponding to previous bins, and adding to the result the lower anomaly of pdf vector i . This can easily be demonstrated by the fact that ADC and DAC errors in lower bins always cause a positive and a negative anomaly of equal magnitude, yielding zero sum. The ADC error on code transition level i causes a higher anomaly in pdf vector $i - 1$ and an opposite lower anomaly in pdf vector i , again yielding zero sum. The DAC error of bin i causes a lower anomaly in pdf vector i , equal to the amount of DAC error, and an opposite higher anomaly in pdf vector i . The latter term is not included in the sum.

Once all DAC errors have been calculated according to this procedure, ADC errors can be derived. The ADC error on code transition level i is equal to the sum of the higher anomaly of pdf vector $i - 1$, and the DAC error of bin i . Again, this can be verified by the fact that the lower anomaly of pdf vector $i - 1$ is equal to the sum of minus the DAC error of bin $i - 1$, plus the ADC error of transition level i .

5.10. Effect of Errors on Overall Integral Non-Linearity

A multi-stage ADC subject to ADC, DAC and gain errors in the first stage, exhibits a very characteristic overall INL curve. Actually, the INL curve provides an alternative to the previous method, for the estimation of errors in the first stage. It is best suited for a quick, visual interpretation, due to the ability of the human mind to efficiently recognize patterns in a figure, despite the presence of

noise. However, it will be seen that the limit histogram method using pdf vector notation has advantages when the limit histogram needs to be reconstructed from an actual histogram, obtained using an ADC with non-ideal stages after the first one.

The limit INL curve of a typical 3-bit first stage is shown on figure 57. The curve is equal to the integral of the limit DNL curve, which in turn is equal to the limit histogram curve, minus 1. Inspection of the curve reveals several characteristic effects, which will briefly be described, without strict proof (the proof would consist of investigating the effect of different errors on the limit histogram curve, and the effect after subtraction of 1 and integration).

- Within each bin, the INL curve is approximately linear. However, all the linear sections have a certain slope. This slope reflects the interstage gain error. A positive slope indicates a gain error ϵ_A that is greater than 1 (gain too large). A negative slope indicates a gain error smaller than 1 (gain too small). The numerical value of the slope is a dimensionless number (lsb/lsb), which can be calculated in either overall lsb ($1\ lsb = R/2^N$) or first stage lsb ($1\ lsb = R/2^M$, with N the overall number of bits, M the number of bits of the first stage and R the input range).
- The INL curve exhibits spikes at the interface between two bins. These spikes are caused by ADC errors. Spikes pointing upward indicate positive ADC errors (comparator reference too high). Spikes pointing downward reflect negative ADC errors (comparator reference too low). The size of the spike reflects the magnitude of the ADC error, in lsb (defined for the overall converter, depending on what its resolution is).
- The linear portions of the INL curve start at different heights. The height of the curve at the beginning of a bin (expressed in overall lsb) reflects the DAC error of that bin. If the linear section within a bin starts off below 0, there is a negative DAC error, otherwise, there is a positive DAC error.

5.11. Measurement of the Limit Histogram

In practice, there is only a limited number of subsequent stages, which only have a finite total number of bits. If the overall converter has N bits (M of which are generated by the first stage), the overall histogram will only have 2^N

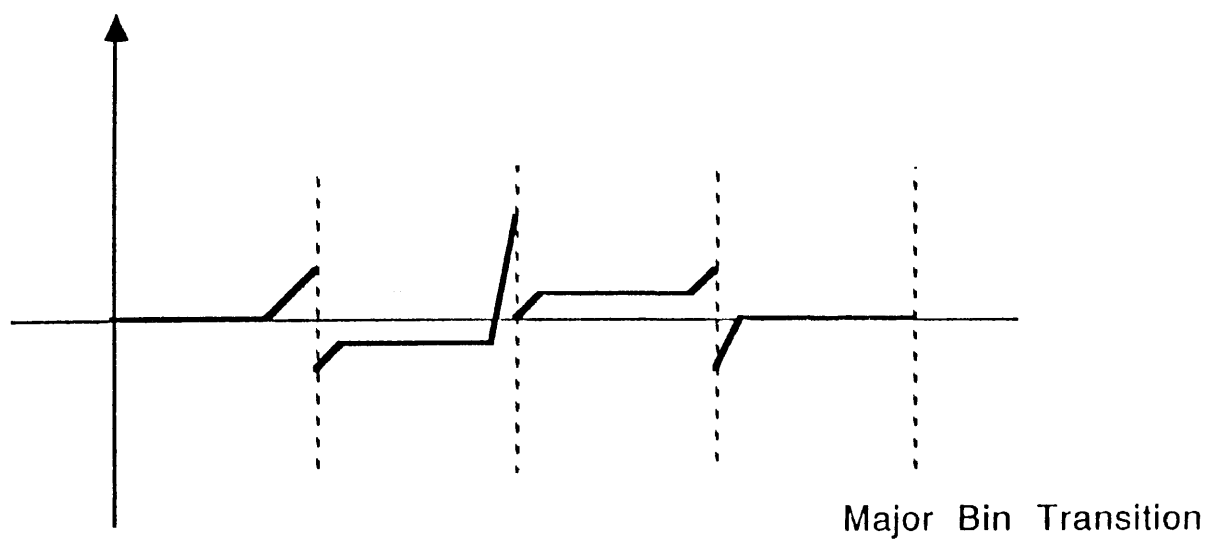


Fig. 57: Typical 3-bit INL Curve

discrete points, instead of having a continuous domain like was assumed for the limit histogram curve. As a result, peaks and gaps will not have the ideal shape that was previously assumed.

Figure 58 shows the typical shape of a histogram with peak and gap. The gap extends over one or more bins, while the peak is localized in one bin. The area of peak and gap can easily be determined, expressed in overall *lsb*. The constant part of the histogram (the area of a bin that is not affected by a peak or a gap) reflects the gain error of the first stage. It is nominally equal to 1, but can be slightly more or less, depending on the actual value of the interstage gain.

In practice, a histogram test may be performed in a slightly noisy environment. The noise in the measurement has the property of "smoothing" sharp histogram edges, like shown in figure 59. This may cause the histogram peak to extend over more than one bin, while the gap may extend over several bins that do not have exactly zero sample count. In such case, area of peak and gap can still be estimated fairly accurately, by integration of the actual histogram values over the affected bins (shaded areas of figure 59).

An additional problem affecting real histogram data, is that the remaining stages of the ADC may not be ideal like was assumed so far. As a result, the normalized histogram values within each bin will not exhibit the typical behavior of one peak or gap, a constant level and another peak or gap. Instead, the constant level will exhibit a slight variation, due to the errors in subsequent ADC stages (figure 60).

In general, these variations are small (much less than 1 *lsb*), due to the fact that errors in subsequent stages affect the overall transfer curve to a lesser extent than errors in the first stage. In addition, these errors tend to be zero mean, just like was the case for the effect of errors in the first stage. This means that the three pdf vector values reflecting the behavior of each bin of the first stage, can still adequately be extracted from real-life histogram data.

First, each major bin (corresponding to the first stage) of the actual histogram (after normalization) is divided into three distinct sections: lower anomaly, constant pdf and higher anomaly. This is shown in figure 60. Next, the average value over the constant pdf section is used as the second dimension in the pdf

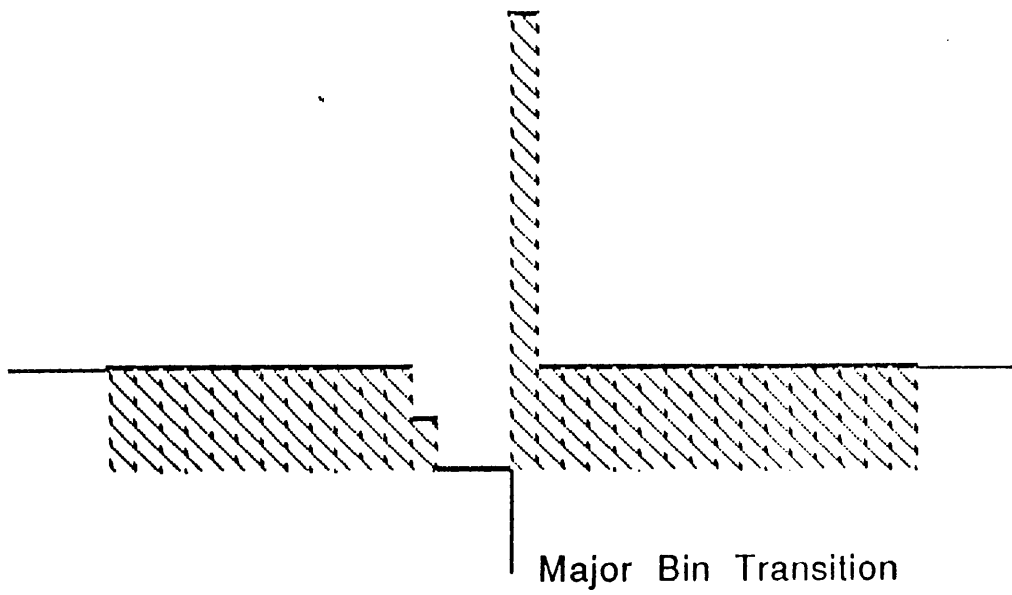


Fig. 58: Peak and Gap in a Typical Histogram

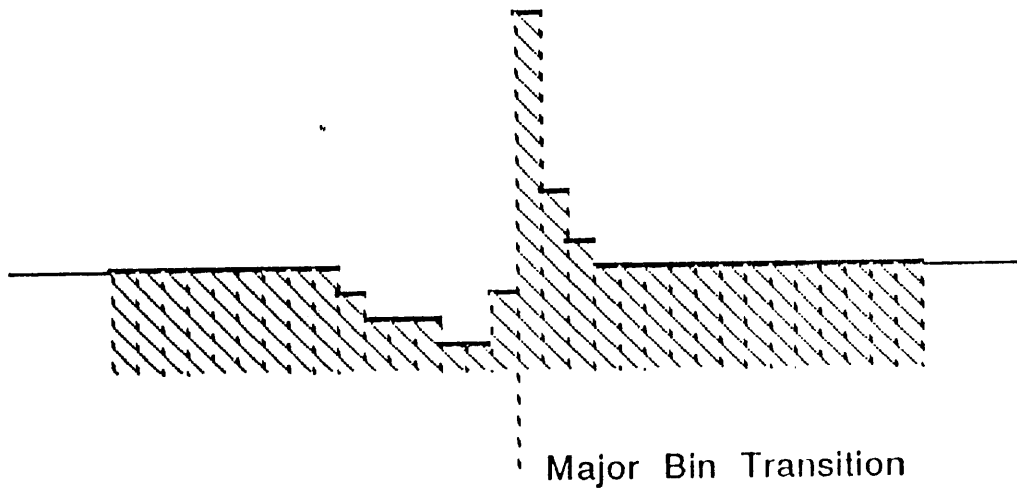


Fig. 59: Smoothing Effect of Noise

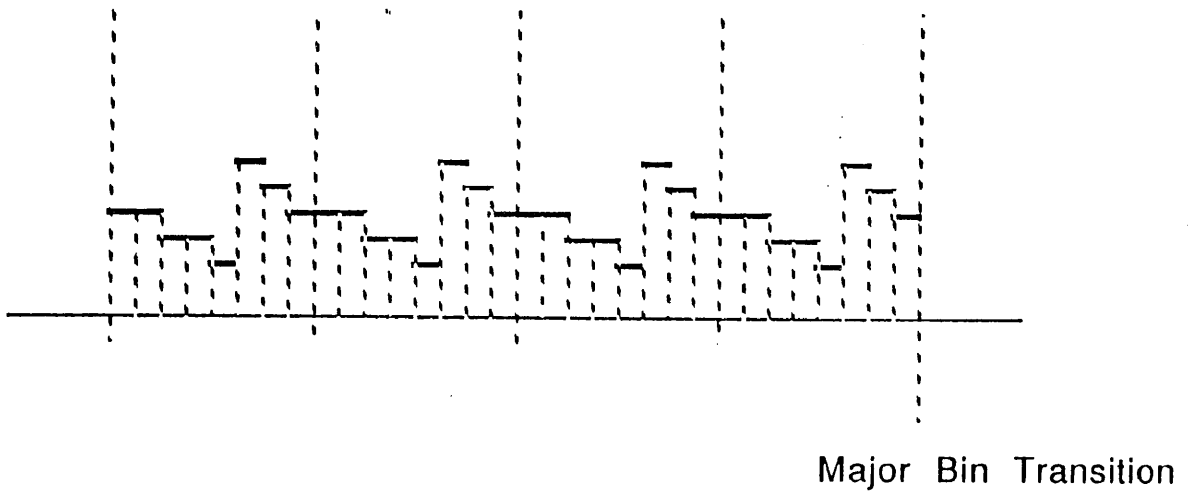


Fig. 60: Histogram Affected by Non-Ideal Subsequent Stages

vector. Finally, the areas of gaps and/or peaks are estimated, relying on knowledge of the constant level. These values are used to describe lower and higher anomalies. Once all pdf vectors are identified, derivation of the individual errors of the first stage can be carried out as described above. The whole procedure has been successfully computerized, into a program that takes a histogram data file and produces a file with a description of the ADC stage.

5.12. Analysis of Errors in Subsequent Stages

So far, we have only described a procedure to identify errors in the *first* stage of a multi-stage ADC. Although these errors are usually dominant (this will be shown in detail in the next chapter), it is often useful to identify errors in subsequent stages as well. This can be accomplished using a technique, very similar to the one used for identification of the errors in the first stage.

To identify the errors in the second stage, it is sufficient to choose a bin of the first stage, that has a peak at the lower as well as the higher end. This can easily be determined from the overall histogram, since the major transition levels (the ones associated with a transition in the first stage) are known. The presence of the peak at both ends of the bin indicates that the range of the residue coming from the first stage exceeds the input range of the next stage from both sides, when the input signal falls into that bin (figure 61).

If the interstage amplifier following the first stage is linear, and the probability density function of the input signal is constant (which has been assumed throughout this discussion), the probability density of the residue from the first stage will be constant as well. Its range will exceed the input range of the second stage, like is required in order to perform a histogram test. As a result, the portion of the total histogram which reflects the bin in question of the first stage, will actually be a valid histogram, reflecting the $N - M$ bits of the converter formed by the second stage and subsequent stages. The same procedure as was used to determine the errors of the first stage, can be used to determine the errors of the second stage, after normalization of the partial histogram. The same procedure can be repeated over and over for additional stages.

Figure 62 shows the measured histogram curve of a commercial multi-step, sub-ranging ADC. In this case, the ADC is composed of two stages of four bits each

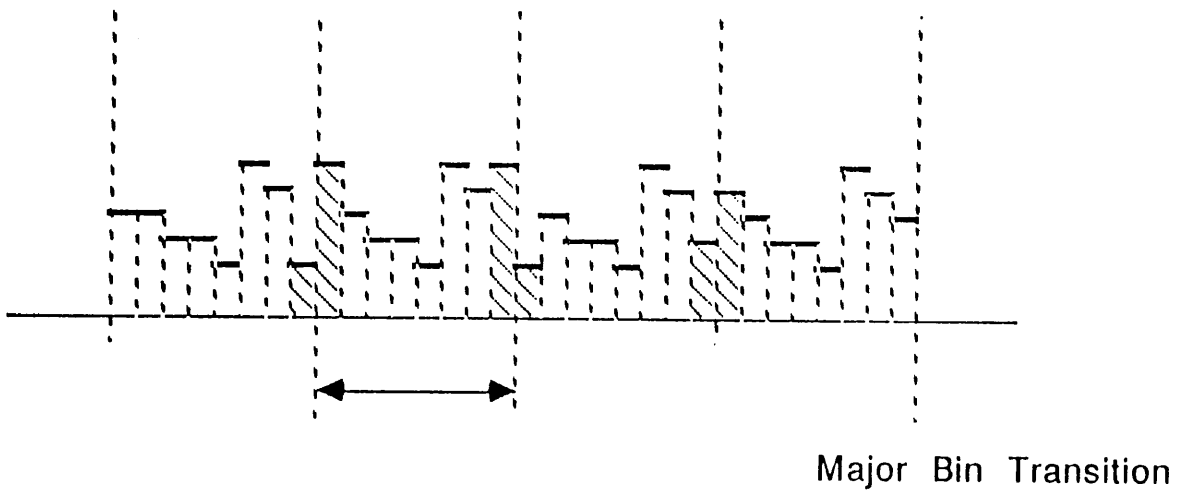


Fig. 61: Bin with Two Peaks

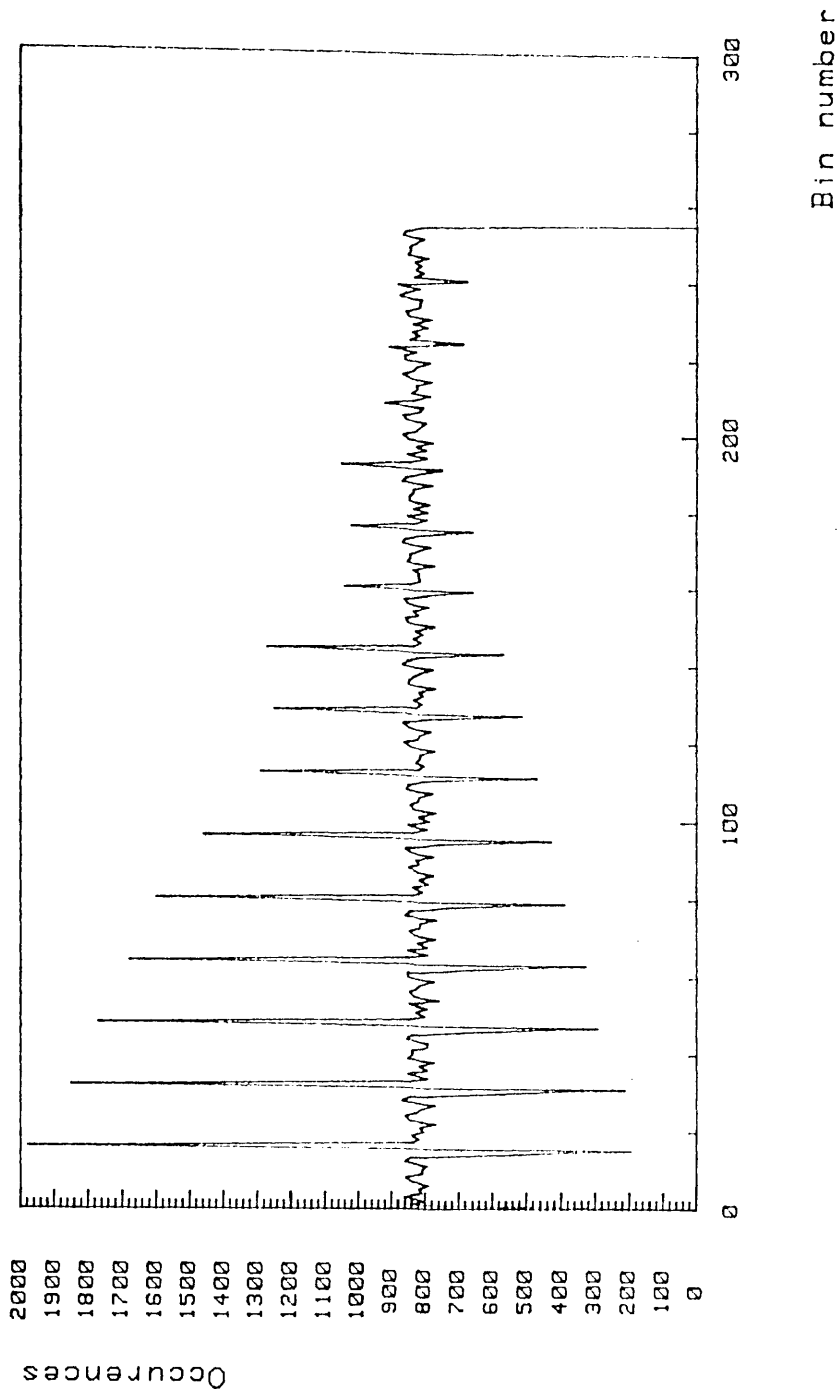


Fig. 62: Measured Histogram of a Commercial Converter

(2 x 4 bit architecture). The histogram was measured using a ramp input signal of 37 kHz, of which 130,000 samples were taken at a rate of about 1 MHz. Figure 63 shows the associated INL curve, obtained after normalization of the histogram, subtraction of 1 (DNL) and partial summation. The measured histogram was also used to determine the errors of the first and second stage, according to the procedure developed above.

In turn, these errors were used to simulate the actual operation of the ADC in a custom, algorithmic ADC simulation program. Such programs have been used extensively for this research, and more simulation results will be discussed in chapter IX. The program used to obtain the data presented in chapter IX is listed in appendix A. The output codes of the simulator were tallied, in order to reconstruct a histogram. The INL curve corresponding to this simulated histogram is shown in figure 64. Like can be seen, the measured and simulated INL curves track each other extremely well (the maximum difference is about $0.3 \text{ } lsb$), which demonstrates the validity of the theory for real-life ADC's.

5.13. Digital Error Correction

It has been demonstrated that due to the limited input range of the second stage of a multi-stage ADC, errors in the first stage can create peaks or gaps (missing codes) in the overall histogram of the converter. In turn, these gaps and peaks reflect discontinuities in the overall INL curve of the converter. In particular, ADC errors in the first stage can create localized, sharp INL peaks, like can be seen on the INL curve of the commercial converter (figure 63).

A very effective technique to limit this effect, is often referred to as "digital error correction" [27, 28, 29, 30]. However, the term "redundancy", which is used as well, is more accurate. The precise theory will be discussed in a lot more detail in the next chapter. However, a short description is appropriate at this point. Redundancy refers to the fact that the input range of the next stage is intentionally designed to be larger than the residue (output) range of the present stage. This avoids situations in which the residue of one stage could be "clipped" when an error in the present stage modifies the range of the residue that is passed on to the next stage.

This safeguards the converter against accumulation of certain errors. In

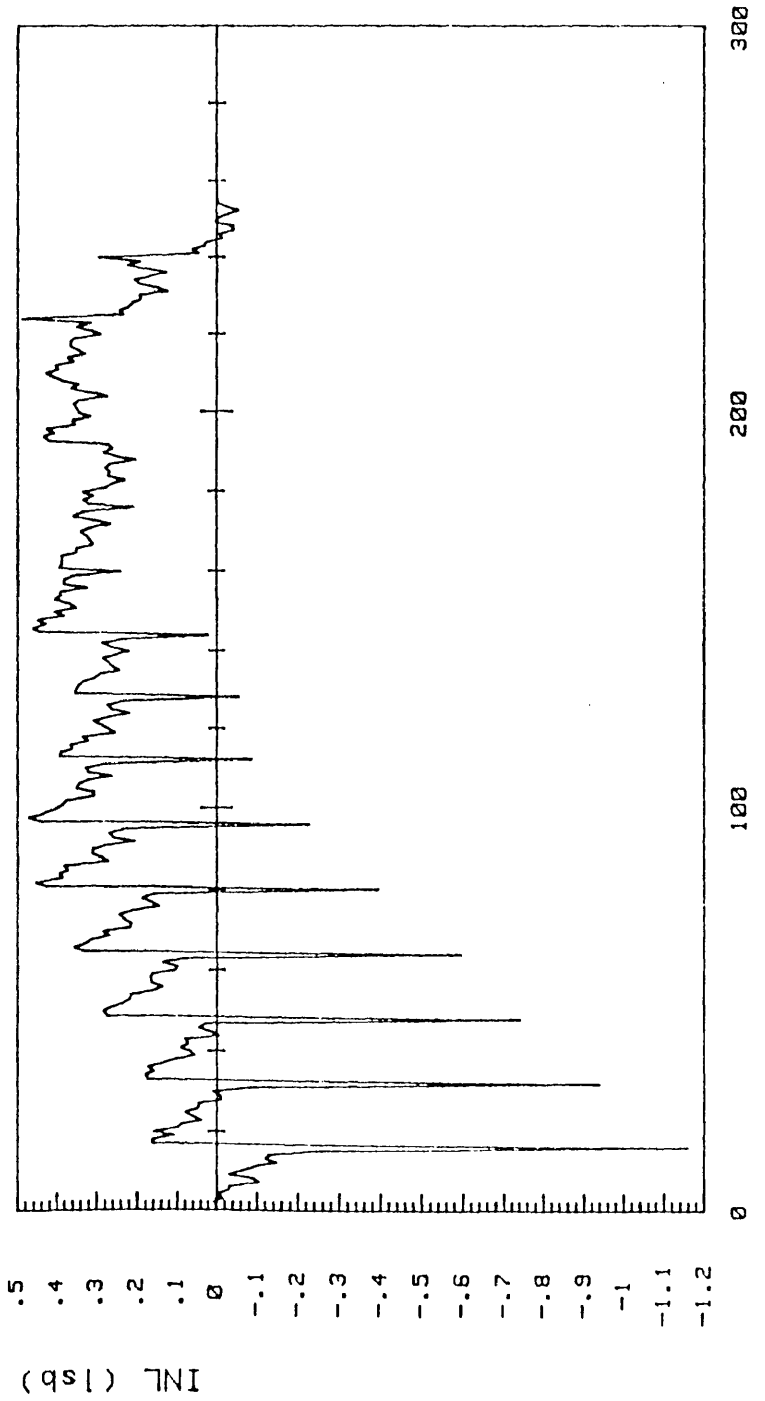


Fig. 63: Measured INL Curve

Transition

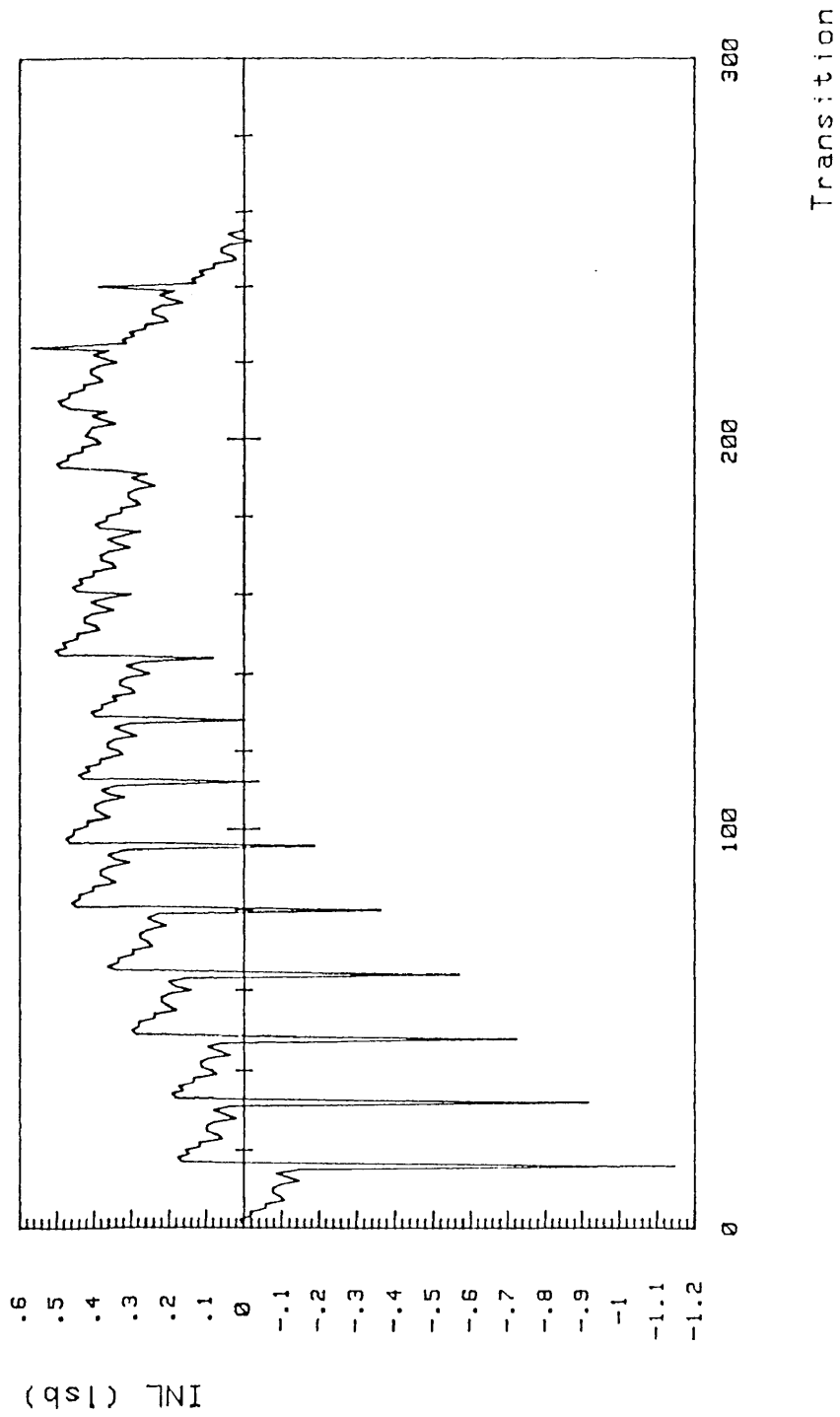


Fig. 64: Reconstructed INL Curve

particular, such converters cannot exhibit peaks in their pdf function, but only gaps. This is due to the fact that if the residue slightly exceeds its nominal range, the over-range capability of the next stage will still make accurate conversion possible. Graphically, using the conventions introduced earlier in this chapter, this could be visualized as if each bin of a stage were slightly larger than nominal, in a way that would cause successive bins to overlap. Whenever two bins overlap, the pdf (or normalized bin width) of the overall converter, at the point of overlap, is given by the sum of individual pdf's. This is opposed to the earlier assumption that whenever a residue would exceed its nominal range, it would be "clipped" (localized infinite pdf or delta impulse).

The redundancy can make the converter highly insensitive to ADC errors, which would otherwise be very hard to control. ADC errors normally cause a gap in the limit histogram, on one side of the transition level, and a peak on the other side. Redundancy allows the excess pdf of one bin to stretch out into the next bin, and fill up the gap, rather than causing a delta peak. This is illustrated on figure 65. It is clear that if the flash error of a stage is less than the amount of redundancy on the input range of the next stage, the overall INL curve will not be affected.

DAC and gain errors are still of concern, since they are not characterized by adjacent pdf peaks and gaps of equal magnitude, like flash errors. As a result, redundancy will not be able to fully compensate for these errors (although it may slightly smooth out their effect). Compensation of this kind of errors will require more sophisticated digital techniques, which will be introduced in chapter VI.

As an example of how redundancy can improve the performance of multi-step ADC's, figure 66 shows the INL curve of the commercial converter discussed earlier, modelled under the assumption that redundancy were used in the second stage, which eliminates the effect of flash errors in the first stage. Had it been used in the actual circuitry, the technique would have brought the ADC within spec (less than 0.5 *lsb* INL error).

5.14. Dynamic Errors

The theory we have developed so far refers only to the static transfer curve of ADC's. This is the transfer curve that would be obtained if the ADC operated

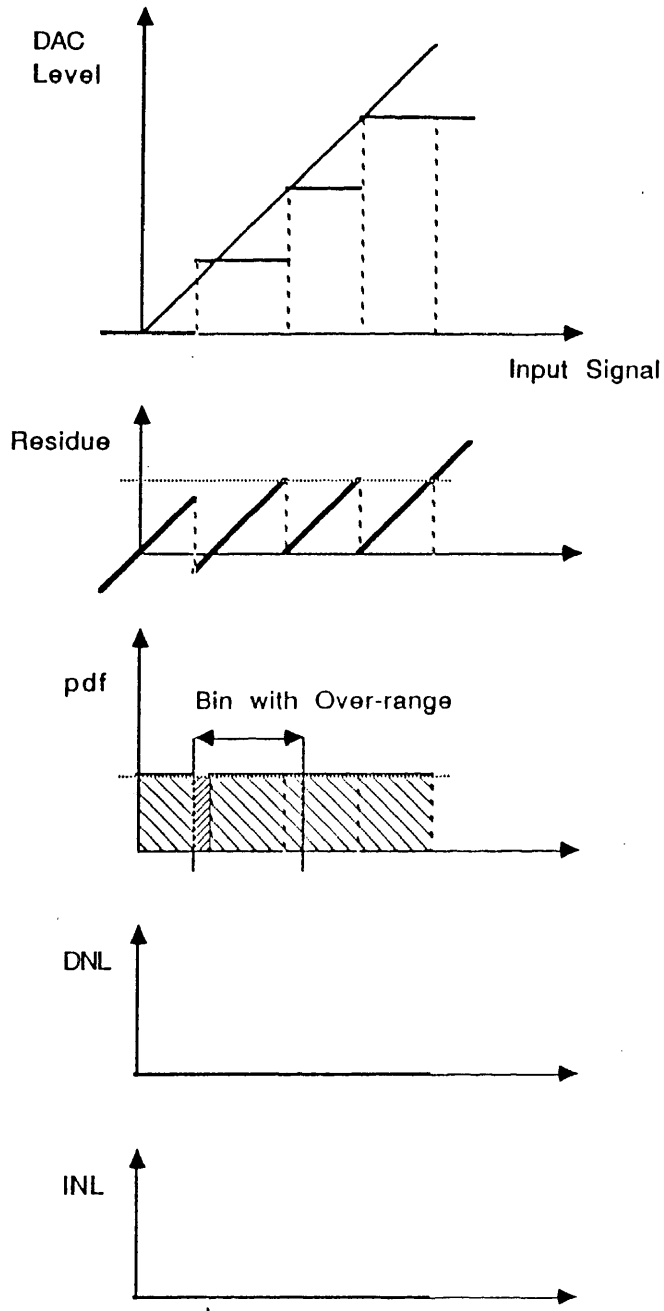


Fig. G5: Stage with Flash Error and Over-Range

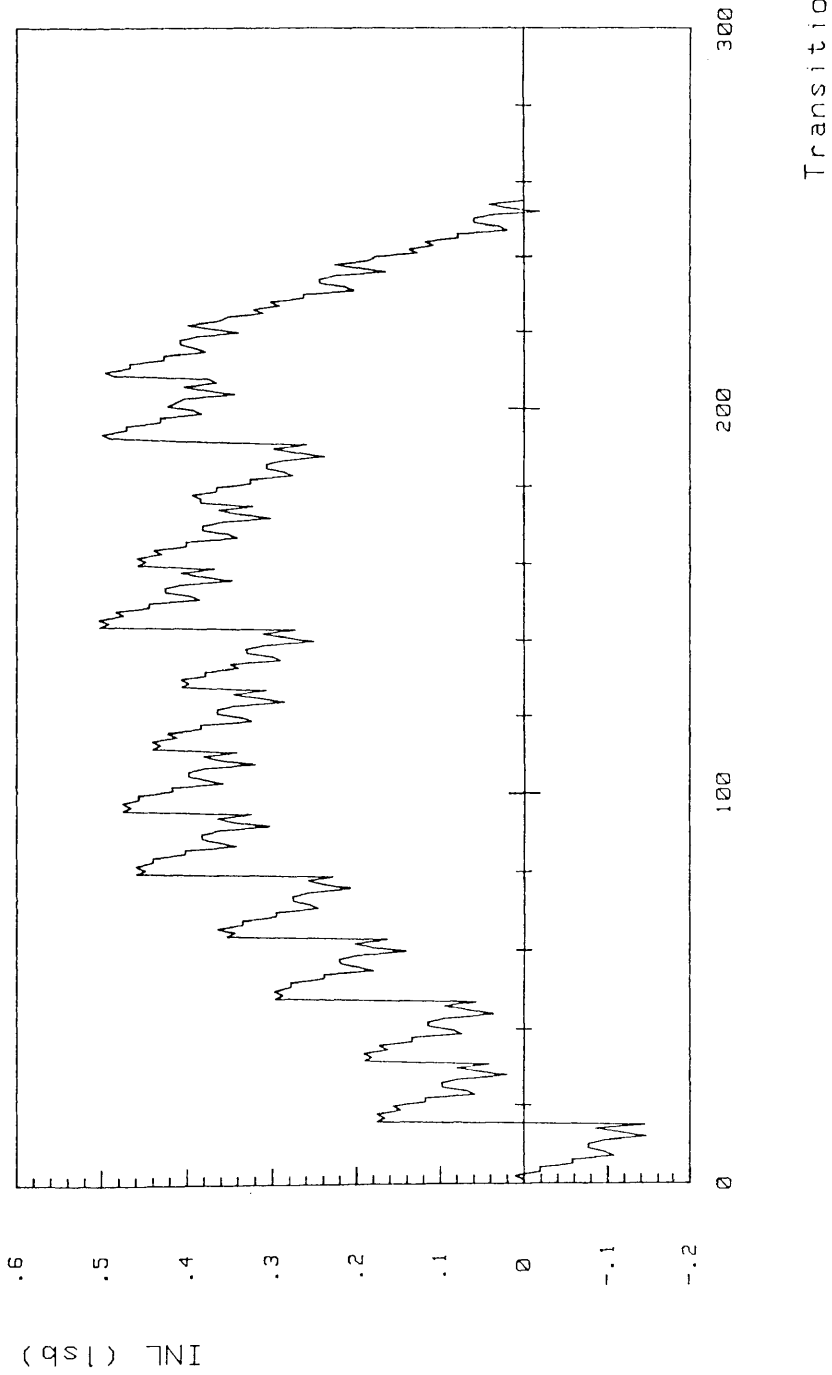


Fig. 66: INL Curve with Error Correction

at low sampling rates, *and* if the input signal were a slowly varying signal. Only under those conditions can we use the simplified, time-independent model we have introduced for one stage.

For a fixed sampling rate, the different errors that can occur within one stage, can be modeled fairly accurately, at least for low-frequency input signals. If the input signal frequency is increased, dynamic errors may start having an effect upon the error model. In particular, effective comparator levels may shift, the value of the interstage gain may vary, or a stage may exhibit a certain amount of hysteresis. These effects are difficult to describe in a complete, general way. However, the dynamic histogram method that was introduced in an earlier chapter, could be used to investigate these effects further. Eventually, future research could produce an adequate, dynamic model of one stage, of which all the variables (errors) can be derived through processing of the dynamic histogram.

5.15. Conclusion

This chapter elaborated on a high-level model for multi-stage analog to digital converters. It described how the transfer function of each stage is influenced by flash (ADC), DAC and gain errors. The effect of these errors was analyzed from a probability density point of view, which was related to the histogram of a converter which would have a non-ideal first stage, followed by ideal other stages. It was shown that from the histogram of such hypothetical converter, all errors of the first stage could be derived. It was shown how the same information could be obtained from a real, non-ideal histogram test. It was shown how other stages could be characterized as well. Finally, error correction techniques based on redundancy were introduced, and an extension of the techniques from this chapter towards dynamic errors was suggested.

CHAPTER VI

DIGITAL ERROR CORRECTION IN PIPELINED CONVERTERS

6.1. Introduction

One of the inherent problems affecting the design of high-performance analog/digital converters is the fact that a highly accurate device is to be made from much less accurate circuit components. Different architectures have been proposed, which try to get around the accuracy problem by either self-calibration (tuning of the critical circuit elements to their ideal values with the help of analog techniques [45, 27, 28, 29, 30, 48, 31, 33, 20, 26, 63, 64, 23, 34, 35, 37, 39, 36]) or digital error correction (compensation of analog errors, using digital circuitry [27, 28, 29, 30]).

The first class of techniques runs into fundamental limitations because no matter how cleverly laid out, analog circuitry can only achieve a limited accuracy. Fully digital error correction has the potential to overcome this limitation, since in the digital domain no fundamental limit on resolution or dynamic range exists. However, digital error correction techniques can only be applied to correct for known errors in the analog signal path. Those errors still have to be estimated somehow, which tends to require a device that is at least as accurate as the device being designed.

This chapter describes a new technique for full digital error correction of pipelined converters. The technique relies on a generalization of the pipelined converter principle that was introduced before, in the sense that comparator thresholds, gains and DAC levels in the pipeline stages do not have to be entirely accurate anymore. As long as the actual values of these parameters are known, digital circuitry can successfully be used to compensate the effect. In chapter VII, a powerful system identification technique will be presented, which allows precise, on-chip determination of the required digital coefficients.

6.2. Generalized Pipelined Converter

A multi-stage, pipelined analog/digital converter (ADC) can be modeled in a general way as a number of pipelined stages, which are connected in series.

Figure 67 is the schematic of one such stage. Each stage has a sample/hold amplifier which periodically samples the input signal and holds a constant value while a local conversion is being performed. The S/H action, when properly clocked, makes the pipelined (parallel) operation of all stages possible, increasing the overall throughput of the converter.

It should be noted that for this discussion, it was found convenient to model the sample/hold amplifier at the input of each stage, rather than at the output, like in previous chapters. Again, the distinction is purely for easy mathematical modeling and does not represent any loss of generality. Similarly, this discussion will use terminology that implicitly assumes that the signal-carrying variable throughout the converter is a voltage. It is obvious how the situation could be generalized for devices in which the signal is carried by other quantities (current, pulse width...).

The gain of the S/H, when correctly dimensioned, ensures a consistent signal level throughout the converter, thus minimizing dynamic range and noise problems. In addition, it makes it possible to build a pipelined converter from identical stages, which optimizes design resources. Depending on whether inverting or non-inverting interstage S/H amplifiers are used, the gain can be positive or negative. The gain of the first stage is usually chosen to be unity.

The local conversion is performed by a number of comparators, which compare the (amplified) input signal to a number of fixed reference voltages (the local ADC levels of that stage). Since all comparators operate in parallel, maximum conversion speed is obtained. The bank of comparators, with associated digital latches, implements an elementary flash converter, which is detailed in figure 68.

The digital pattern of comparator outputs is applied to a rudimentary D/A section. Depending on the pattern, this section generates different output voltages (the local DAC levels). The DAC output voltage is subtracted from the amplified input signal in order to generate a local residue, which is passed on for conversion by the next stage. A simplified, switched-capacitor implementation of the subtraction, gain, sample and hold actions is given in figure 69. The operation of such gain stages will be discussed in detail in chapter IX. A more symbolic representation of one stage is given in figure 70. Figure 71 shows how the stages

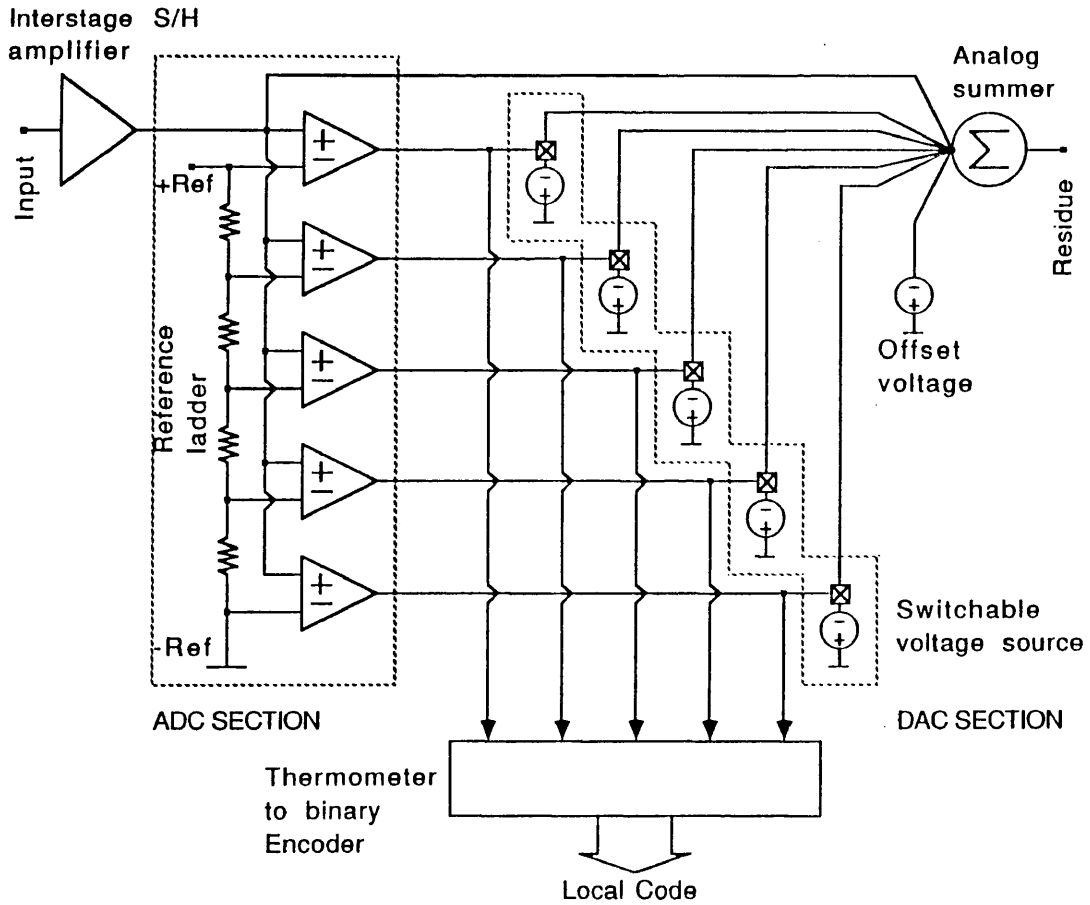


Fig. 67: Schematic of a Pipeline Stage

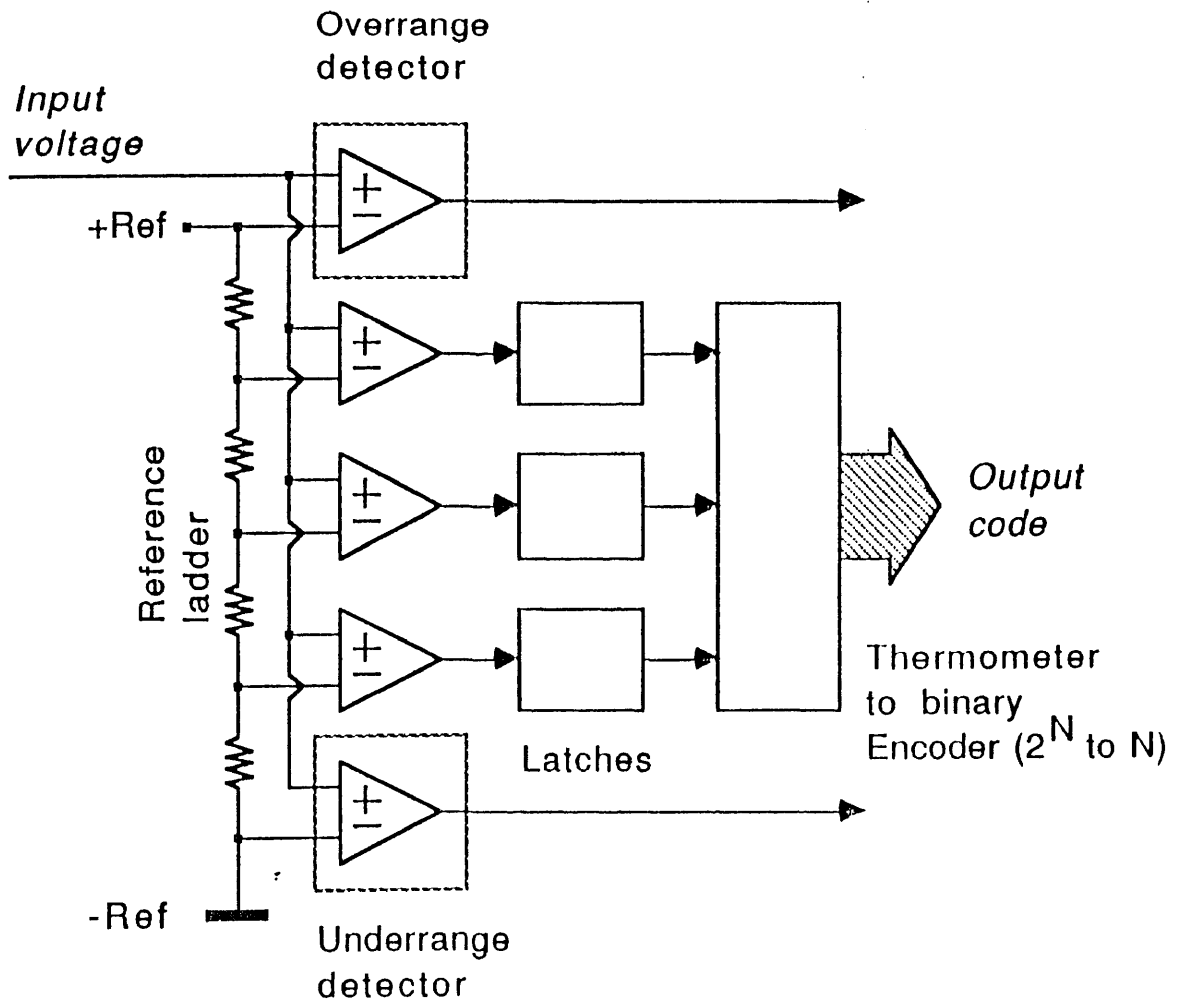


Fig. 68: Flash Converter

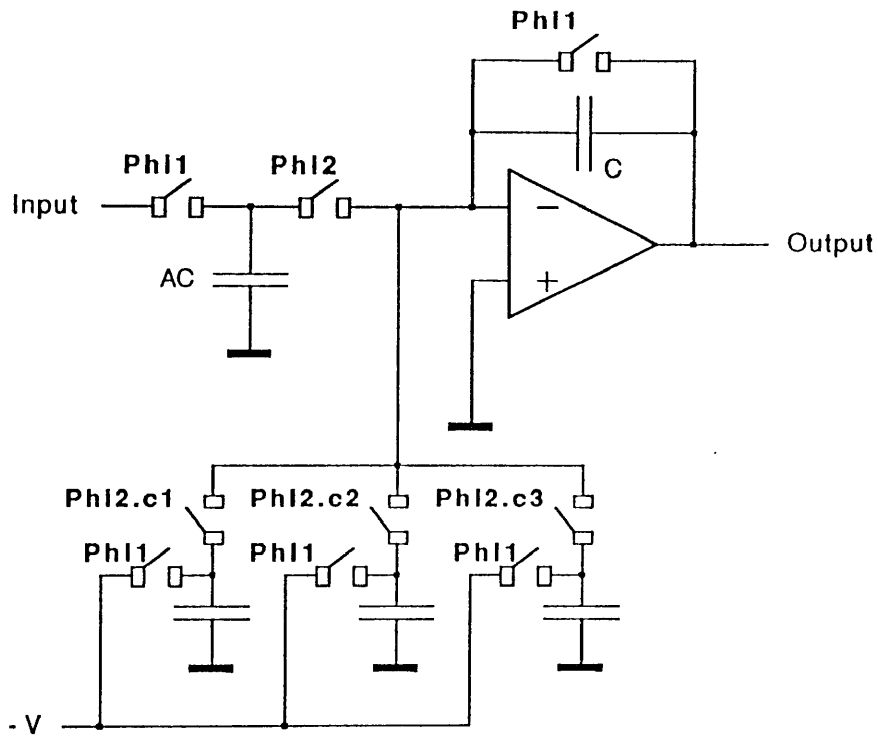


Fig. 69: Stage with Switched-Capacitor S/H

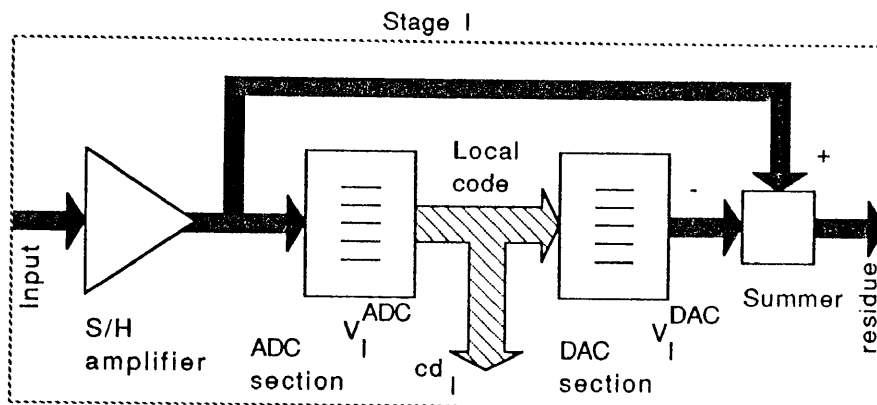


Fig. 70: Symbolic Representation of a Stage

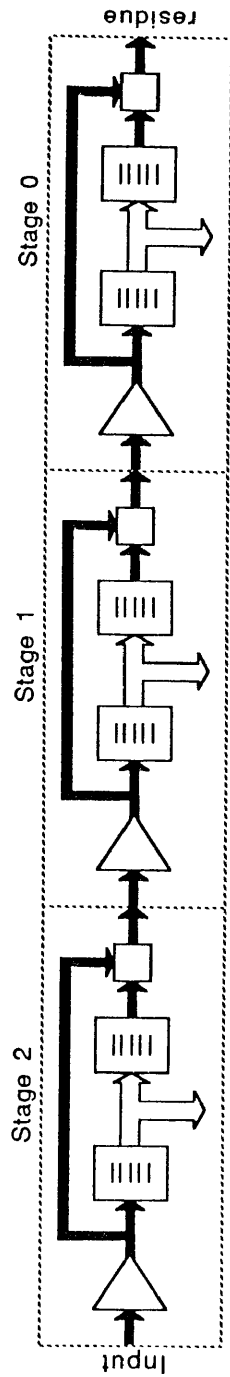


Fig. 71: Cascaded ADC Stages

are cascaded.

We will assume that the pipeline is composed of L stages, numbered from 0 to $L - 1$. The symbol l will be used to refer to a particular stage. $l = L - 1$ will designate the first (input) stage; $l = 0$ will designate the last stage. Each stage has a number of comparators, referred to by the symbol m or m_l . The total number of comparators in stage l will be represented by the symbol M or M_l .

The M comparators of stage l generate a thermometer-type output pattern, which can be interpreted as a digital code. There are $M + 1$ possible codes, which we will call local codes and to which we will assign the values $0 \cdots M$. We will refer to these values with the symbol cd_l . The value of the local code is a function of the input voltage to stage l , based on the local ADC transition levels $[V_l^{ADC}(0) \cdots V_l^{ADC}(M - 1)]$. This can be expressed by the equation

$$cd_l(V_l^{in}) = \begin{cases} 0 & \text{for } V_l^{in} < V_l^{ADC}(0) \\ 1 & \text{for } V_l^{ADC}(0) \leq V_l^{in} < V_l^{ADC}(1) \\ \dots & \\ M & \text{for } V_l^{ADC}(M - 1) \leq V_l^{in} \end{cases} \quad (64)$$

We will further refer to this equation as the *ADC relation* of stage l .

This expression can only be evaluated if the input signal to a stage is known. This input signal is the residue of the previous stage (except at the input), and can be quite hard to estimate. However, a lot of conclusions about the structure and accuracy of a pipelined ADC, can be reached without knowledge of the precise values of cd_l .

The input voltage of stage l , V_l^{in} , is amplified by the input S/H amplifier with gain A_l , which we will initially assume to be linear. The input amplifier section may have a certain input-referred offset voltage V_l^{off} . Depending on the local binary code cd_l , a certain DAC output voltage V_l^{DAC} is subtracted from the amplified signal in order to calculate the local residue res_l . For any particular cd_l , a general equation can be written to express the relationship between the residue and the input signal of stage l .

$$res_l = (V_l^{in} - V_l^{off}) \cdot A_l - V_l^{DAC}(cd_l) \quad (65)$$

The expression can be rearranged in order to yield the input signal as a function of the local code cd and the local residue res_l .

$$V_l^{in} = \frac{res_l + V_l^{DAC}(cd_l)}{A_l} + V_l^{off} \quad (66)$$

This illustrates the meaning of the local code cd_l . DAC level cd_l , scaled by the gain of the input amplifier, is an approximation of the local input signal and the local residue is the approximation (or quantization) error. The pipelined ADC algorithm is based on amplifying this residue and converting it again in the next stage. Gain and ADC and DAC levels are chosen in such a way that for normal input voltages, the residue would be within the input voltage range of the next stage. It will be shown that this makes the influence of the residue in the approximation of the input voltage decrease after each stage.

The global quantization error of the pipelined ADC is due to the residue of the last stage ($l = 0$), which is not converted any further. This residue is given by the expression

$$res_0 = (V_0^{in} - V_0^{off}) A_0 - V_0^{DAC}(cd_0) \quad (67)$$

From the structure of the pipeline, it is clear that

$$V_{l-1}^{in} = res_l \quad (68)$$

The expression for the input voltage of a stage can be expanded to

$$V_l^{in} = \frac{\left[\frac{res_{l-1} + V_{l-1}^{DAC}(cd_{l-1})}{A_{l-1}} + V_{l-1}^{off} \right] + V_l^{DAC}(cd_l)}{A_l} + V_l^{off} \quad (69)$$

or to

$$V_l^{in} = \left[\frac{V_l^{DAC}(cd_l)}{A_l} + \frac{V_{l-1}^{DAC}(cd_{l-1})}{A_l A_{l-1}} \right] + \left[V_l^{off} + \frac{V_{l-1}^{off}}{A_l} \right] + \frac{res_{l-1}}{A_l A_{l-1}} \quad (70)$$

By further expanding the same expression and evaluating it for $l = L - 1$, the input voltage of the global pipelined ADC with L stages (0 to $L - 1$) can be written as

$$V^{in} = V_{L-1}^{in} = \sum_{l=0}^{L-1} \left[\frac{V_l^{DAC}(cd_l)}{\prod_{j=l}^{L-1} A_j} \right] + \sum_{l=0}^{L-1} \left[\frac{V_l^{off}}{\prod_{j=l+1}^{L-1} A_j} \right] + \frac{res_0}{\prod_{j=0}^{L-1} A_j} \quad (71)$$

or after the substitution $\prod_{j=0}^{L-1} A_j = P$

$$V^{in} = \frac{1}{P} \left[\sum_{l=0}^{L-1} \left[V_l^{DAC}(cd_l) \prod_{j=0}^{l-1} A_j \right] + \sum_{l=0}^{L-1} \left[V_l^{off} \prod_{j=0}^l A_j \right] + res_0 \right] \quad (72)$$

In this expression, we can define

$$W_l(cd_l) = \frac{1}{P} \left[\left[V_l^{DAC}(cd_l) \prod_{j=0}^{l-1} A_j \right] + \left[V_l^{off} \prod_{j=0}^l A_j \right] \right] \quad (73)$$

The expression for V_{L-1}^{in} now reduces to

$$V^{in} = \sum_{l=0}^{L-1} W_l(cd_l) + \frac{res_0}{P} \quad (74)$$

We will further refer to this equation as the *DAC relation*.

W_l is the weight associated with the conversion of a signal in stage l . It represents one term in the DAC relation. We will simply call it a *DAC weight* of that stage. Strictly speaking, this weight has the dimension of a voltage. The particular W_l value of a given ADC stage l depends on the specific input voltage V_l^{in} to that stage, but only through the local code cd_l . This code can only take a few discrete values $[0, 1 \cdots m]$, as determined by the ADC relation (equation (64)). However, considered individually, the M_l DAC weights of stage l [$W_l(0), W_l(1) \cdots W_l(m)$] are input-independent and time-invariant parameters of the pipelined ADC.

The practical application of the pipelined ADC consists of using the DAC relation to calculate an approximation for the input signal V^{in} . The residue of the last stage (res_0) is discarded and the resulting error is considered the inherent quantization error of the conversion. Since the main constraint on an ADC is usually linearity, rather than precise gain or offset, we could tolerate a scaling constant or an offset on the W_l .

In many, if not most traditional converters, the interstage gain values A_j of equation (73) are designed to be integer values which are powers of two. This

tremendously simplifies the implementation of the DAC relation (equation (74)), since a binary division by two can be accomplished by shifting a number of digital lines within the circuit, and the addition of different numbers can be performed by combining digital lines, without the need for elaborate digital adders. However, the theory developed here does not rely on this fact at all, and any value of the gain could be used if appropriate digital logic is provided to implement the DAC relation.

In simpler pipelined converters, the W_l values are often implicitly scaled so that each one of them would represent an integer value (usually unsigned and binary weighed), with the smallest one of them being equal to unity. The W_l are then considered to be dimensionless. Since their sum represents the ADC output (the conversion result), they are often referred to in *lsb* units. The conversion result (CR) is then given by

$$CR(V^{in}) = \sum_{l=0}^{L-1} W_l'(cd_l), \quad W_l' \epsilon Z \quad (75)$$

with the cd_l given by the ADC relation.

Independently of the exact absolute value of the weights, the fundamental idea is that each conversion consists of the addition of L weights (one corresponding to each stage), which are each selected from a bank of $M_l + 1$ possible choices (one choice corresponds to one possible local code cd_l in stage l).

6.3. Redundancy

In a practical pipelined ADC, enough inherent redundancy can always be built into the different stages to make the global structure insensitive to a certain amount of error on the ADC levels. The process has been described in a previous chapter. It consists of extending the input range of the next stage with respect to the output (residue) range of the current stage. As a result, the residue of one stage can never overdrive the input of the next stage (not even in the presence of a limited amount of errors within the stage), and no non-linearity will be introduced due to pdf peaks (chapter V). This technique has conventionally been known as digital error correction. However, this kind of correction does not change anything

to the fundamental principle of the pipelined ADC; it simply guarantees that successive residues do not grow beyond certain bounds.

If in a stage l , the ADC code cd_l is not determined accurately, the residue may become larger than intended. By extending the range of the ADC section of the next stage however, the residue can still be adequately converted, and the new residue is forced back to smaller values. This can be accomplished by adding one or more extra (redundant) comparators and associated DAC sections to that stage, to detect an over-range or under-range condition. The mechanism prevents the residues from growing beyond the linear range of the components. The DAC levels associated with the redundant codes, can be characterized by their weights W_l , just like regular DAC levels.

This error correction scheme does not require extra comparators in each and every stage of the pipeline. If a worst-case estimate can be made for all component errors (ADC levels, DAC levels, gains and offsets), the worst-case variation on all residues can be determined, and error correcting (redundant) comparators can be added only to those stages where the accumulated error could otherwise allow the residue to exceed a certain safe range. In practice, this means that in the whole pipeline, only a few redundant comparators can adequately keep the residue under control.

An alternative to using extra comparators, is to reduce the gain of the interstage amplifiers in order to insure that the amplified residue would be smaller than the nominal range of the flash (comparator) section following the amplifier. Since the theory developed here does not rely on particular values of the gain, it is even possible to use non-integer nominal gain values.

The only restriction is that the gain would be greater than 1, since otherwise, the overall quantization error of the converter (last term of the DAC relation (74)) could increase to a large value, instead of becoming negligible. Increasing the number of stages while maintaining interstage gains larger than unity increases the factor P ($\prod_{l=0}^{L-1} A_l$), by which the last residue is divided in equation (74). As a result, the global quantization error $\frac{\epsilon_{90}}{P}$ can be made arbitrarily small. This guarantees an upper bound on the global quantization error of the pipelined ADC, for any useful input signal.

6.4. Calibration Procedures

Redundancy does not improve the accuracy of the DAC levels or associated weights W_l . It merely guarantees that all residues will be within bounds, and that the whole circuit operates in its linear range. As such, the technique has the potential to eliminate the influence of incorrect ADC levels. In the absence of any additional error correction scheme, this shifts the accuracy requirements away from the ADC sections, towards the DAC sections, which tend to be easier to make accurate.

However, if the W_l values used in the conversion do not match the actual DAC levels, a conversion error, independent of the quantization error, will still be made. It is clear that if the circuit parameters within the ADC are not very well controlled, this error will limit the accuracy much more drastically than the quantization error, which can be made arbitrarily small by adding stages. Any effective error correction scheme should concentrate on making the W_l match the actual circuit parameters, or vice-versa. This process is called calibration. It is clear that there can exist two classes of calibration procedures:

- Techniques that aim at forcing circuit components to their nominal values, calculated to obtain matching to certain pre-determined W_l values. Examples are feed-back systems and analog calibration loops.
- Techniques that aim at leaving the values of analog circuit components unaffected, but instead update the digital W_l values in order to reflect the analog components.

6.5. Estimation of the DAC Weights

In practice, the precise (should-be) values of the weights W_l are not known. The analog sections of the pipelined converter are dimensioned in order to implement certain *target* values of W_l . In the absence of any better control mechanism, those target values are used as approximations for the actual W_l . ADC circuits are usually dimensioned so that the W_l would be binary weighed integer values, which makes the combination of the local codes cd_l into the global conversion result easier, as described earlier.

Depending on the particular circuit technique used to implement the DAC sections (capacitor arrays, switched current sources etc.), a discrepancy in the

order of 0.1% to 1% may exist between the target values and actual values of W_l . This error can be reduced by laser-trimming or analog self-calibration techniques, which fall under the first class of calibration procedures described above. They focus on forcing the W_l to their target values. The problem of such techniques is that they only work to a certain accuracy, which is determined by analog components and extremely hard to bring down.

The other approach, corresponding to the second class of calibration procedures, consists of accurately measuring the W_l and storing them, in digital form, into RAM memory. During the normal operation of the pipeline, the local codes cd_l can be used to select the RAM location associated with a DAC weight, in the fashion of a look-up table. An L -input digital adder can be used to combine the table values in accordance to the DAC relation, in order to calculate the conversion result in a real-time fashion. Due to the nature of the pipelined ADC, the addition can actually be performed with pipelined digital logic as well, like illustrated in figure 72 (one stage) and figure 73 (cascade of three stages).

This is essentially different from trying to force the W_l to target values. We will refer to this calibration procedure as *full digital error correction*. Since digital hardware is used to account for the W_l , there is no fundamental limit to the achievable resolution or accuracy. The technique only relies on the assumption that all gain sections of the pipeline are linear and that all components are time-invariant.

Of course, the main problem remains the accurate evaluation (measurement) of the analog components within the ADC, in order to determine the correct W_l values. This could theoretically be accomplished using precision measurement equipment. However, it would only displace the problem, since the measurement equipment would need superior accuracy. In the next chapter, an alternative technique will be described, which allows extremely accurate characterization of the W_l values, using only the converter itself.

6.6. Truncation Error

In a practical pipelined ADC, the *actual* W_l are not precisely binary weighed, nor are they integer values. This poses additional problems as far as the accuracy of the conversion result is concerned, even when assuming that we could measure

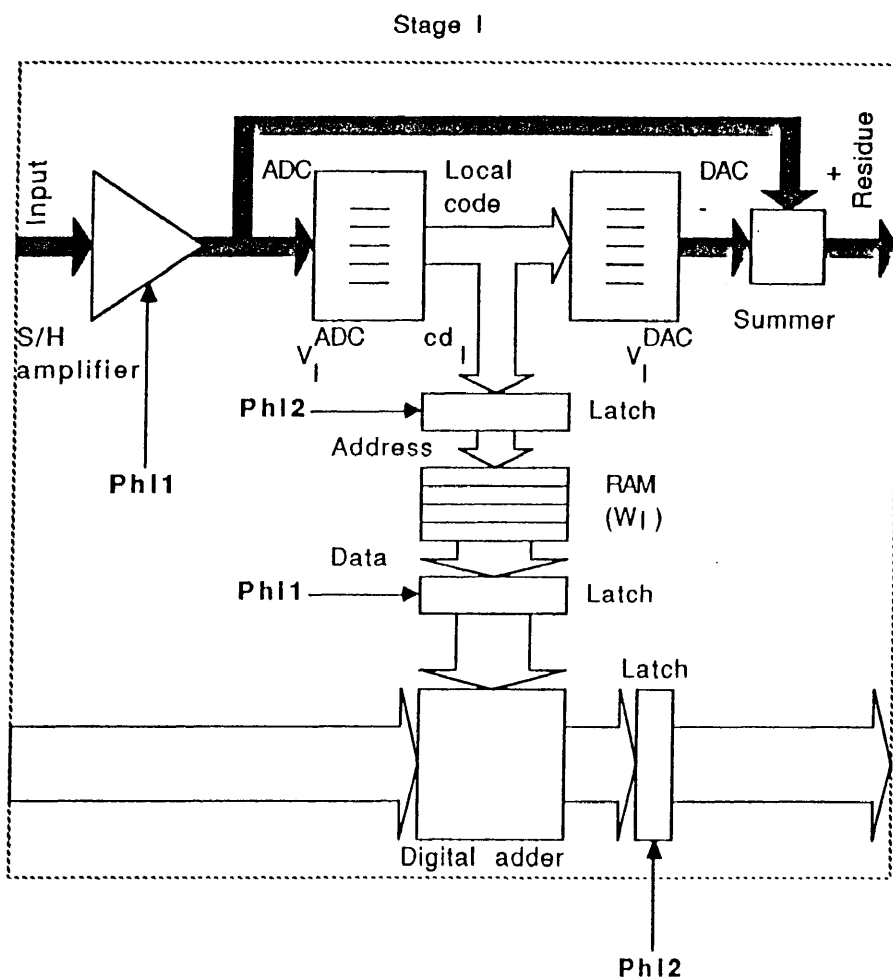


Fig. 72: Stage with Pipelined Arithmetic Section

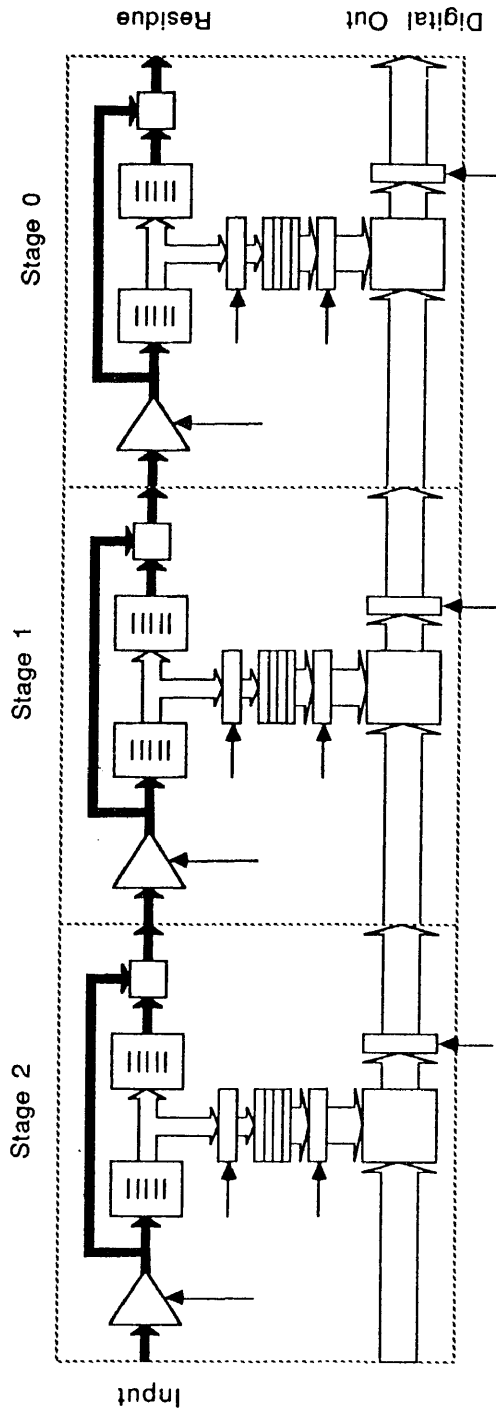


Fig. 73: Three Stages with Arithmetic Section

all W_i with infinite resolution. As opposed to the idealized expression of equation (75), the exact theoretical conversion result is not an integer number and cannot be represented by a limited number of bits. Since eventually the ADC output has to come in a finite, binary form, the sum of the terms in the DAC relation, must be rounded or truncated.

This introduces an additional error on top of the inherent quantization error of the pipeline, and on top of the error on the W_i themselves. However, this effect can be limited to acceptable levels by proper accuracy and error management (number of bits used in the look-up tables and throughout the different sections of the pipeline).

6.7. Monotonicity

A minor drawback of full digital error correction, is that the resulting ADC is not guaranteed to be strictly monotonic. There are situations where a very small increase in input signal level produces a decrease in the ADC output or vice-versa, no matter how accurate all the weights W_i are known.

The global pipeline has a large number of code transition levels, which are defined as the boundary voltages separating two distinct conversion results. Every time the input voltage crosses such a transition level, one of the local codes cd_i in one of the stages flips to a different value, and the residue of that stage makes a discontinuous transition. It is clear that this will result in a sudden transition of the last residue, which is the quantization error.

Since the quantization error is bound, and small, the corresponding jump of the conversion result will be small as well. If the conversion result is truncated to a certain number of bits, chances are that the jump will not even be noticeable. But if a transition level coincides with a truncation boundary, a small non-monotonicity will show up in the ADC transfer characteristic.

In practice, the pipeline will be designed so as to keep the quantization error well below ± 0.5 lsb, and the non-monotonicity will be insignificant. Conventional evaluation techniques which rely on monotonicity, like the histogram test (for calculation of differential and integral non-linearity) can still be applied.

6.8. Dimensioning of the Interstage Gains

The S/H amplifier of the first stage in the pipeline normally has unity gain, while all the other (interstage) gain sections of the circuit have a gain determined by the particular organization of comparators and DAC levels in the previous stage.

Each stage of the pipeline can have any number of comparators, M , as long as there is at least one. In practice, the number will be kept small in order to minimize total chip area, power consumption and sensitivity to component errors. The threshold voltages of the M comparators will form the ADC levels of the particular stage. Each stage is limited by a specific useful signal range, within which sufficient linearity is guaranteed. We will refer to this range with the symbol R . A pipeline stage usually is designed so that the M ADC levels divide the useful voltage range R , available for the ADC, into $M + 1$ equal sections. That way, each section corresponds to a particular local code cd , and the coverage of the codes is spread evenly.

The DAC levels of a stage usually are spaced apart by the same amount, which minimizes the possible voltage range of the residue. If both the ADC and DAC levels have equal spacing ΔV , which is equal to $R/(M + 1)$, the range of the residue will also be limited to ΔV . The offset on the DAC levels is chosen so that the residue, after amplification, will match the useful ADC voltage range of the next stage. This means that the gain of the next stage should be equal to $M + 1$. If redundancy is to be included in the next stage, the gain can be made accordingly smaller.

6.9. Extension for Non-Linear Gain

So far, it has always been assumed that all gain sections of the pipeline were linear. In practical circuits, provided adequate care is taken, this assumption may hold up to the 12 or 13 bit level. Beyond, non-linearity will certainly limit the application of full digital error correction in its elementary form. However, the same principle can be extended to provide compensation for a certain amount of gain non-linearity, at the expense of added circuit complexity.

The DC transfer curve of a non-linear S/H amplifier (figure 74) can always be approximated by a piece-wise linear curve (figure 75). The more linear sections are used in the model, the better the approximation. We will refer to the number

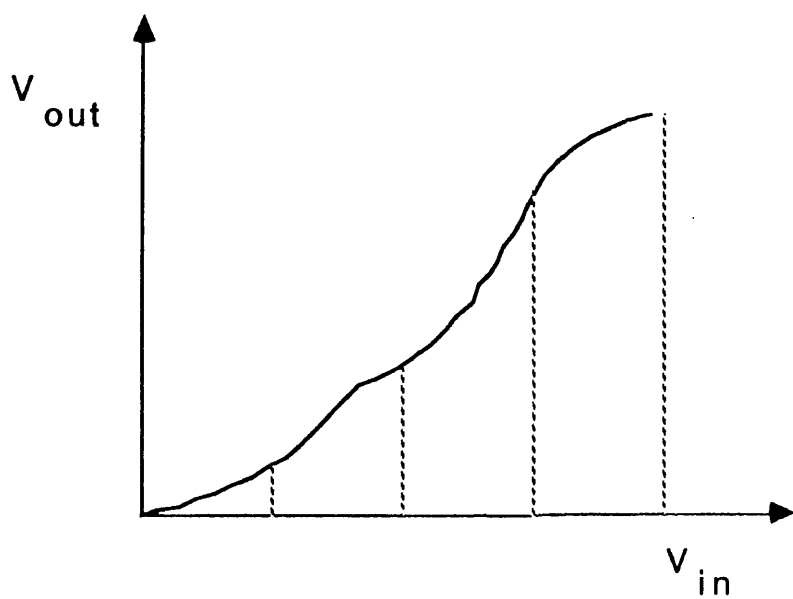


Fig. 74: Non-Linear Gain Characteristic

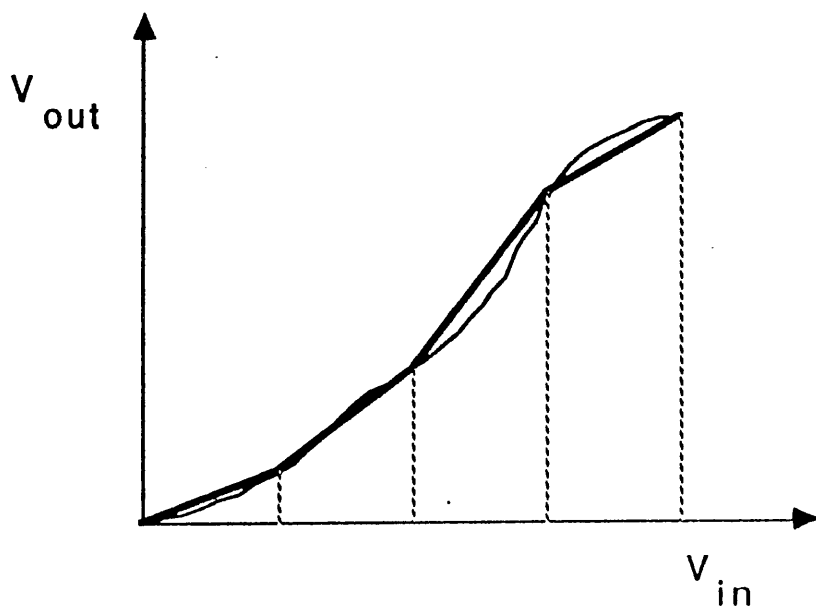


Fig. 75: Piece-Wise Linear Approximation

of linear sections used to model the transfer curve of the amplifier of stage l with the symbol K_l .

All the formulas derived to model a pipelined ADC and leading to the DAC relation, still hold for piece-wise linear gain stages, provided that all gain and offset values used in the computation of the DAC weights W_l are changed according to which specific linear sub-section of each amplifier the signal travels through.

Unfortunately, like can be observed from the DAC relation, the W_l values of each pipeline stage contain a factor in their denominator, which corresponds to the gain of each of the preceding stages. This means that in order to correct the W_l of a stage, the local codes and precise gain values of the preceding stages have to be taken into account, which may be impractical.

However, a pragmatic approach can be taken. The non-linearity of an amplifier can be modeled as the injection of a parasitic voltage to the signal within that amplifier. If the non-linearity of the amplifier is considered to be piece-wise linear, the parasitic voltage will also be piece-wise linear, and will reflect the difference between the linear amplifier characteristic and the piece-wise linear approximation (figure 76).

The non-linearity of the S/H amplifiers, modeled as an injection of parasitic voltage within a stage, can also be seen as an input-dependent error on the effective DAC levels of a stage, which means an input-dependent error term on the DAC weights W_l of that stage. According to the DAC relation, the associated overall error will be most critical in the first stages of the pipeline. This means that for weak non-linearities, one can effectively correct for amplifier non-linearity by only considering the first few stages of the pipeline.

In general, it is fairly easy to make a worst-case estimate of the relative non-linearity in the S/H amplifiers. With the help of the DAC relation, one can easily estimate how many of the front stages need to be corrected. In a similar way, one can estimate how many segments need to be used for the piece-wise approximation of the actual transfer curve of each amplifier, in order to keep the overall error due to an imperfect piece-wise linear approximation, sufficiently small.

The correction can then proceed as follows. We could consider the S/H amplifier of stage l , and assume that its transfer curve can be approximated

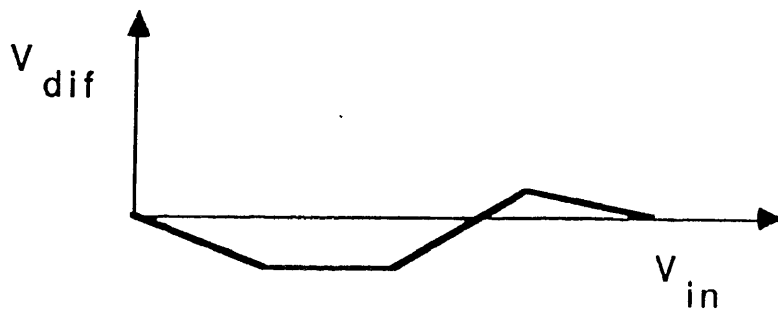
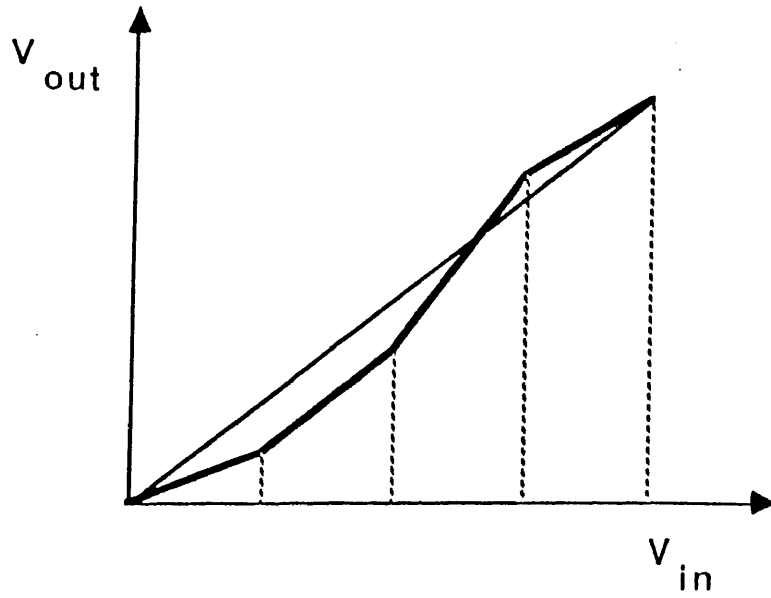


Fig. 76: PWL Approximation Error

closely enough by K_l linear segments, in such a way that one end point of the first and the last segment coincides with one end point of the linear, endpoint calibrated transfer curve, which has a corresponding gain of A_l . We will assume that the range of input voltages corresponding to each segment, is the same (equidistant linear segments). This is the situation of figure 75. The piecewise linear approximation can be fully characterized by K_l gain values A_l^k (for k ranging from 0 to $K_l - 1$), which correspond to the local gain values of each segment. One can easily verify that in that case, the following relation holds:

$$A_l = \frac{\sum_{k=0}^{K_l-1} A_l^k}{K_l} \quad (76)$$

The A_l^k values can be fairly accurately determined by a first-order measurement procedure. It is enough to apply two input voltages, which are as close as possible to the breakpoint voltages of the different segments, measure the difference in amplifier output voltage and divide it by the difference in input voltage. This measurement could be performed using the pipeline itself. In chapters VII and VIII, a system identification technique called "Accuracy Bootstrapping" will be introduced. This technique allows iterative identification of a system, starting out from the uncalibrated system.

Once the K_l A_l^k values are determined, the input-dependent correction terms for the V_l^{DAC} of stage l can be calculated, according to figure 76. We will use the dimensionless variable x ($0 \leq x \leq 1$) to refer to the input voltage to the amplifier, in such a way that $x = 0$ corresponds to the lower end of the input voltage range and $x = 1$ to the higher end. We will use a similar dimensionless variable y to refer to what part of a linear segment a particular input voltage corresponds to.

The segment number k , corresponding to x , is given by

$$k = \text{int}(K_l x) \quad (77)$$

The position y within segment k is given by

$$y = \text{frac}(K_l x) \quad (78)$$

The correction term corresponding to x , $c(x)$, can then be written as

$$c(x) = \left(\sum_{i=0}^{k-1} \frac{A_l^i - A_l}{K_l} + y \frac{(A_l^k - A_l)}{K_l} \right) R \quad (79)$$

With R the input voltage range of the amplifier.

A drawback of compensating only some stages in this way, is the fact that the regularity of the whole pipeline structure is disturbed, which in turn makes it harder to design. The hardware evaluation of expressions (77) and (78) can be simplified considerably by choosing K_l equal to a power of 2, since then the multiplication by K_l and integer or fractional part determination reduces to a rearrangement of bit lines. However, the evaluation of expression (79) requires the use of at least one multiplier for each linearity-compensated stage. A multiplier has the disadvantage of being more bulky than an adder.

Any practical pipelined ADC scheme is likely to be determined by a trade-off between desired accuracy and circuit complexity. (The basic digital correction scheme can be implemented with relatively little additional hardware.) There is no significant *fundamental* limit to the accuracy achievable with a digitally compensated pipelined ADC, even when using slightly non-linear S/H amplifiers.

However, it should be stressed that this is only true to the extent that the differential DAC levels in the system are signal-independent, and that the summing operation is linear. In practice, these assumptions may not be justified anymore in systems designed for extremely high accuracies. In chapters VII and VIII, Accuracy Bootstrapping will be introduced. This technique will make it possible to determine the DAC weights necessary to linearize the overall transfer function of a pipelined converter. It can achieve dramatic improvements in linearity, but the underlying implicit assumptions will be the same: the DAC levels, as well as the summing operation must be time-independent and linear. (In any kind of self-calibrated system, there must be some reference; in this case, the linearity reference is formed by the linearity of the summing process.)

A method to compensate for the non-linearity using a piece-wise linear approximation of the actual characteristic was outlined. However, this is not the only possible scheme. If additional logic can be tolerated (in some cases the logic may even be readily available, e.g. as part of a micro-processor based system), more sophisticated compensation schemes could be conceived. Instead of a piece-wise linear approximation of the amplifier characteristic, a polynomial approximation could be used, based on the estimation of a few points of the characteristic. However, additional research would be needed in order to make

the idea practical.

6.10. High-Speed Operation

The digital correction technique was primarily designed for the cancellation of static errors. However, due to the nature of the pipeline, it will work just as well at high clock rates or signal frequencies, as long as one can assume that signals are effectively sampled on clock transitions and then held constant during the duration of each conversion cycle.

The digital logic within an integrated ADC system can easily be made fast enough for operation at several tens of MHz. The same thing can be said of the comparators, which need to be fast, but not necessarily very accurate. A practical limit for high speed operation of the pipeline, is imposed by the characteristics of the S/H amplifiers used in the circuit. The linearity of their transfer characteristic is crucial, but the requirements can be somewhat relaxed by application of the extended, piece-wise linear, accuracy-bootstrapped algorithm.

However, settling performance and timing accuracy of the S/H sections are likely to form the bottom line. The input S/H samples the incoming analog signal and determines input frequency range and overall timing accuracy. Settling performance of interstage S/H sections will further determine the accuracy of the conversion at high sampling rates.

Reliable operation for high sampling rates is only guaranteed insofar each S/H output can settle to the required accuracy (number of bits) in the time allocated (e.g. one clock cycle). The error associated with incomplete settling is not a time-invariant error, since it may depend on the previous value of the input signal as well as on the present value. As such, the error cannot easily be corrected for by fixed digital coefficients. Conceptually, it may be thinkable to also correct for dynamic errors in a digital way, but the required logic is likely to become prohibitive.

6.11. Noise

The cascade of gain stages throughout the pipelined ADC may create noise problems. Noise in the analog signal path cannot be distinguished from the actual signal, will be converted and will eventually show up as digital noise on the output

code of the ADC. The noise problem can get worse for high-frequency operation of the pipeline, since the bandwidth of all S/H amplifiers has to be made large. This increases the total integrated noise generated in these amplifiers.

In each stage, noise is mainly generated by two effects: the S/H amplifier and the noise voltage on the DAC voltage sources. However, to simplify the mathematical treatment, both kinds of noise can be combined, and one can define an equivalent input-referred noise voltage for each S/H amplifier. It is clear that noise in the pipeline will be of no concern as long as the effective rms noise voltage on the last residue is smaller than half the range of that residue. If this is the case, the digital noise on the ADC output will not exceed ± 0.5 lsb rms.

Since the last residue is determined by the cascade of the L interstage S/H amplifiers, its effective noise voltage can be expressed as:

$$v_{n,res} = ((v_{n,L-1} A_{L-1} + v_{n,L-2}) A_{L-2} + v_{n,L-3}) A_{L-3} + \dots \quad (80)$$

With $v_{n,l}$ the equivalent input-referred noise voltage of stage l .

If all stages can be assumed to have identical S/H amplifiers and equal noise characteristics, this expression can be rewritten as

$$v_{n,res} = v_n (A^L + A^{L-1} + \dots + A^2 + A) \quad (81)$$

This shows that noise can become a problem at high speeds (large bandwidth and hence higher v_n values for a constant noise power spectral density) and for a large number of stages (high number of effective bits).

However, the situation looks more favorable if a certain amount of zero-mean, random noise (non-harmonic distortion) can be tolerated on the conversion result. The structure of the pipeline guarantees that no matter how many stages are present, none of them can be saturated by noise build-up (because of the redundant comparators, which force any excessive residue back within range). During the calibration, the W_1 could still be determined accurately if many measurements are averaged together. That this takes more time, usually is of no concern during calibration.

6.12. Advantages of Fully Digital Correction

Full digital error correction has the potential of eliminating the effect of the following errors in a pipelined ADC: incorrect transition voltages in the ADC sections (e.g. due to a parasitic offset voltage in the comparators or to incorrect reference voltages), incorrect output voltages in the DAC sections of each pipeline stage, offset voltages in the interstage amplifiers, and incorrect gain in these amplifiers. Especially the ability to handle gain and DAC errors is valuable, since these errors are especially hard to control.

Most other approaches attempt to correct for these effects using analog self-calibration techniques, which require high-precision components and sophisticated circuitry. Despite these, the achievable overall accuracy is intrinsically limited by the fixed, inherent accuracy of the calibration circuitry. This forms a fundamental obstacle to the development of high-accuracy (over 10 bits), high-speed (tens of MHz) data converters.

Traditionally, high accuracy could only be obtained using low-speed or oversampled techniques (like the sigma-delta converters). Due to their nature, these techniques cannot reach the high sampling rates possible with a pipelined converter. Full digital error correction, combined with accuracy bootstrapping, the system identification technique to be described in the next chapter), offers the prospect of achieving very high accuracy at the high sampling rates of pipelined ADC's.

Oversampled techniques tend to require complex circuitry. This makes these converters expensive precision devices, which cannot easily be integrated together with other functions, because of their large required chip area. Accuracy bootstrapping has the potential to be much more area-efficient. It requires little components other than the pipeline itself, especially if the calibration controller function can be implemented by general-purpose logic, shared with other components. Since digitally compensated pipelined ADC's can be built on a relatively small chip area, the price of the high-speed, precision data converter function can be expected to be low, and the yield of chips high. It also becomes feasible to integrate such devices as part of larger systems. Alternately stated, because of their much smaller area and high accuracy, accuracy-bootstrapped pipelines could become very competitive in the lower-speed, high-precision range as well.

Since the technique corrects for the dominant types of component errors, the requirements on the IC technology used to implement a pipelined ADC, are eased as well. This offers the possibility to integrate pipelines in relatively standard, high-volume and cheap technologies (like used for digital applications), rather than in specialized, more accurate and more costly analog technologies. The excellent immunity to component errors achievable with accuracy bootstrapping further increases the yield and decreases the cost of high-precision pipelined ADC's.

Accuracy bootstrapping does not have to be limited to pipelined ADC's. It may be possible to apply the same concepts to sigma-delta or other data converters, or even to tunable analog filters.

For completeness, it should also be mentioned that full digital error correction of devices does not *have to* be combined with on-chip self-calibration hardware. Some CMOS integrated circuit technologies provide the possibility for on-chip electrically erasable programmable read only memory (EEPROM). Such memory cells are non-volatile and could be used for in-factory storage of calibration information.

6.13. ASIC Aspects

Because of its inherent simplicity and the regularity of the associated hardware, full digital error correction is very suitable for ASIC (application-specific integrated circuit) design techniques. ASIC's are characterized by a very short design turn-around time, which is often achieved through the extensive use of design automation tools, which generate parametrized, standard cells. This obviously reduces design time and cost.

The regularity of a digitally compensated pipelined ADC occurs in two directions. First of all, all stages of a pipelined ADC, including most of the associated logic, are very similar (or even identical). This means that once one stage is developed, it can be duplicated to yield pipelines of any length (L), and hence any number of bits.

In addition, each stage is made up of a number of basic subcircuits, like comparator/latch combinations (ADC section), switchable voltage sources (DAC section), an interstage S/H amplifier, and some additional logic. It is clear that both the ADC and the DAC section are very regular. They are merely a repetition

of M (number of comparators) identical blocks. This makes it very easy to parametrize a pipeline stage and make M a variable (which does not even have to be a power of 2).

Since the gain of the S/H amplifier is dependent on the number of comparators in a stage, and the gain is set by a resistor or capacitor ratio, it is even possible to have this gain automatically adjust itself to the value of M , by adding more capacitor sections in parallel or resistor sections in series as more comparator/latch/voltage source sections are chosen. The same thing can be said of the reference voltage generator (resistive ladder).

Altogether, using a few basic building blocks, it becomes straightforward to design digitally compensated pipelines with any combination of L and M values. A lay-out could even be generated automatically. The approach becomes even more attractive if the calibration procedure of the pipeline is performed off-chip, in software. This makes the technique a perfect candidate for the implementation of custom high-speed, high-accuracy, low-area ADC's that are to be used as part of a bus-oriented microprocessor system.

6.14. Conclusion

A general mathematical description of pipelined ADC's was given, from which a digital error correction technique was derived. The technique consists of connecting a look-up table to each stage of the pipelined converter. The coefficients in the table digitally represent the possible "weights" of that stage in the total conversion. Theoretical values were associated with the different weights, based on corresponding terms in the expression for the conversion result (DAC relation). The correction technique was extended to the case where the interstage gain sections of the pipeline could be non-linear. The fundamental limits of the approach were investigated for high-speed, high-precision data converters. Finally, a discussion of alternative implementations, advantages and ASIC aspects was given.

Throughout the discussion, it was stressed that although full digital correction can provide superior performance, the overall accuracy of the converter is only as good as the digital coefficients provided. The accuracy bootstrapping principle, described in the next two chapters, will provide the means to determine these coefficients extremely accurately, using only the pipelined ADC itself, and no external calibration standards.

CHAPTER VII

ACCURACY BOOTSTRAPPING

7.1. Introduction

It was shown in chapter VI how most errors in each stage of a pipelined analog to digital converter can be compensated for, using fully digital techniques. The idea was to set up the equation describing the operation of the converter as a function of key component values of each stage. Knowledge of the errors on those values makes it possible to compute coefficients to be placed in a look-up table, thus realizing precise conversion up to any desired degree of accuracy.

Unfortunately, knowledge of the actual errors within the devices is hard to obtain, even with the use of precision measurement or calibration equipment. Accuracy bootstrapping provides a practical but extremely powerful solution to this problem. It is a novel, iterative system identification technique, which determines the value of components within the system, by using parts of the system to measure other parts. In the case of a pipelined converter, it has been shown how dramatic increases in overall linearity could be obtained.

This chapter describes accuracy bootstrapping in entirely general terms, making abstraction of the implementation or even the intended use of the "system". In this chapter, the general system architecture requirements are discussed. Occasional references are made to pipelined ADC's, as a typical example of a system. In the next chapter, the application of accuracy bootstrapping to pipelined ADC's will be described in much more detail.

7.2. General Principles

Accuracy bootstrapping is a technique that makes it possible to design systems, with very accurately control over their transfer function. The transfer function expresses the relationship between a quantity at the system input (input signal) and a corresponding quantity at the system output (output signal).

The exact nature of input and output signals is irrelevant. Although accuracy bootstrapping was originally designed to calibrate electronic systems (primarily

pipelined ADC's), it is not unthinkable to apply the same technique to other systems (e.g. mechanical control loops, scales, various sensors...) In the general sense, the signal could be represented by a voltage, a current, a pulse width, a digital code, a pressure, a temperature, a setting, the position of a control device (steering wheel, lever...) or actuator (hydraulic pump, servo-motor...), a fluid level etc.

Any time such a system is built, the actual transfer function (here simply meaning the relationship between an input and an output) is normally subject to a certain amount of inaccuracy or unpredictability, due to the fact that it is technologically impossible to size every component of the system exactly to its nominal, desired value. There is always a tolerance on the value of resistance, capacitance, weight, size, elasticity constant etc. of components used within any physical system.

Accuracy bootstrapping performs system identification (determining of unknown component values) as well as system calibration (fine-tuning). It starts out with an inaccurate system, and gradually adjusts it until a desired level of accuracy is reached, in other words: until the actual transfer function of the system resembles the desired transfer function closely enough, or until all unpredictability due to random component errors (tolerances) is gone.

7.3. System Requirements

By its very nature, accuracy bootstrapping cannot be applied to just any system. The following conditions are necessary.

- 1 . The system within which the technique is to be applied, must be designed in a very specific way, in order to make the necessary sequence of operations possible. In particular, the system must be composed of one or more blocks (sub-circuits or stages in the case of an ADC), each with at least one input and at least one output. A deterministic relationship (transfer function) must exist between each output and the inputs.

To a certain extent, it must be possible to logically reconfigure these blocks, so that different outputs would successively be connected to certain inputs and vice-versa. Obviously, this kind of reconfiguration must be such that outputs of one block are connected to compatible inputs of other blocks (voltage to

voltage, current to current etc.), with adequate input range. For the purpose of accuracy bootstrapping, it is not necessary that the blocks be similar. In practice however, it is often more convenient to design a system with nominally identical blocks, like a pipelined ADC with interstage gains.

An abstract example of a system with three blocks is shown in figures 77 and 78. The figures show two different configurations. In each case, each block has two inputs and two outputs. Figure 79 shows a configuration which is typical of pipelined converters. The system has two stages, each with an analog and a digital input. The system input is the analog signal to be converted. The analog output of each block is the residue, which is fed to the analog input of the next block. The last residue is discarded. The digital input of each block is the partial conversion result, generated by previous blocks. Each block adds a term to this result, according to the internal, local conversion code and the coefficients in local RAM memory (chapter VI). The digital output of the last block is the overall conversion result.

It is possible (though not necessary), that at some point, the output of one block would be logically connected to the input of the same block. It is equally possible that at some point, several blocks would be connected in a loop. This is the case, for instance, in recycling or partially recycling (hybrid) pipelined converters. Since these are discrete-time systems, a block can hold a signal during one time frame (clock period), while the output is fed back to the input during the next time frame. The effect of such procedure is that the sequence of operations is spread out in time rather than in space. The functionality however, is the same.

2. It must be possible to apply control signals (or settings) to the system in order to determine which block is logically connected to which other block at any given time, e.g. by using controllable switches. By changing the control signals, different configurations are realized within the system. This is shown in figure 80. Accuracy bootstrapping requires several configurations to be realized successively, which is normally accomplished by consecutively applying different combinations of control signals (a control sequence).

The control sequence could be generated by a part of the system, further referred to as the configuration controller (CFC). The control signals could

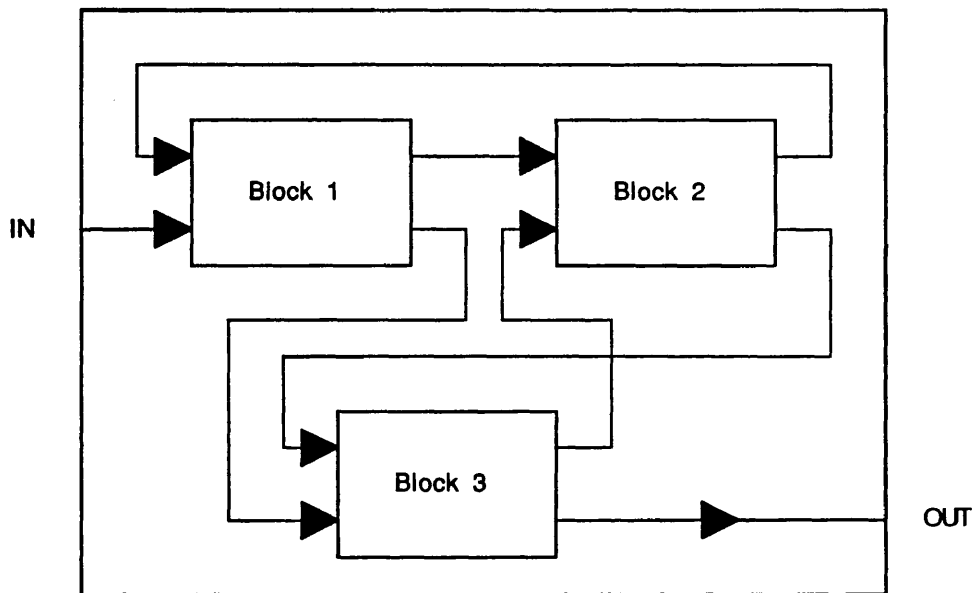


Fig. 77: System Consisting of Reconfigurable Blocks

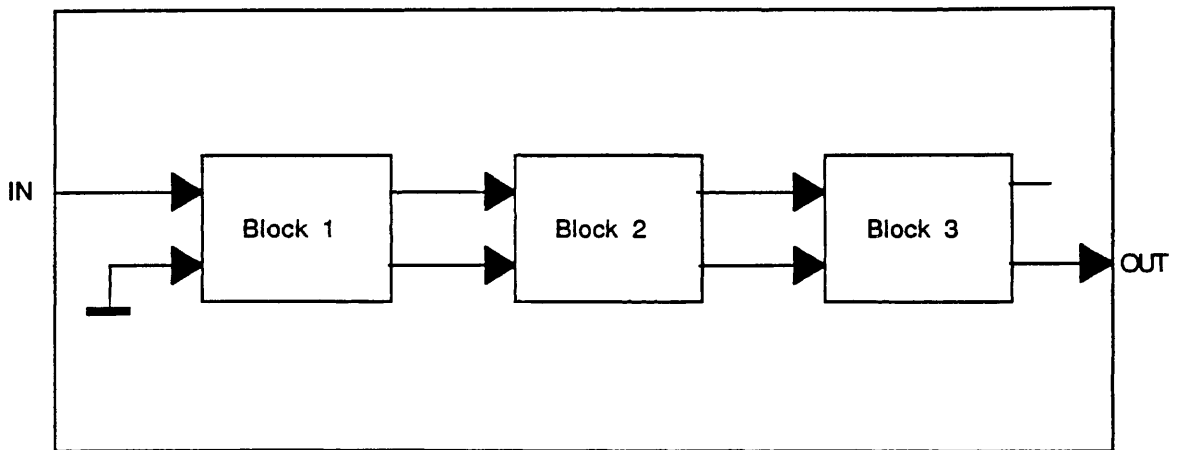


Fig. 78: Different Configuration of the System

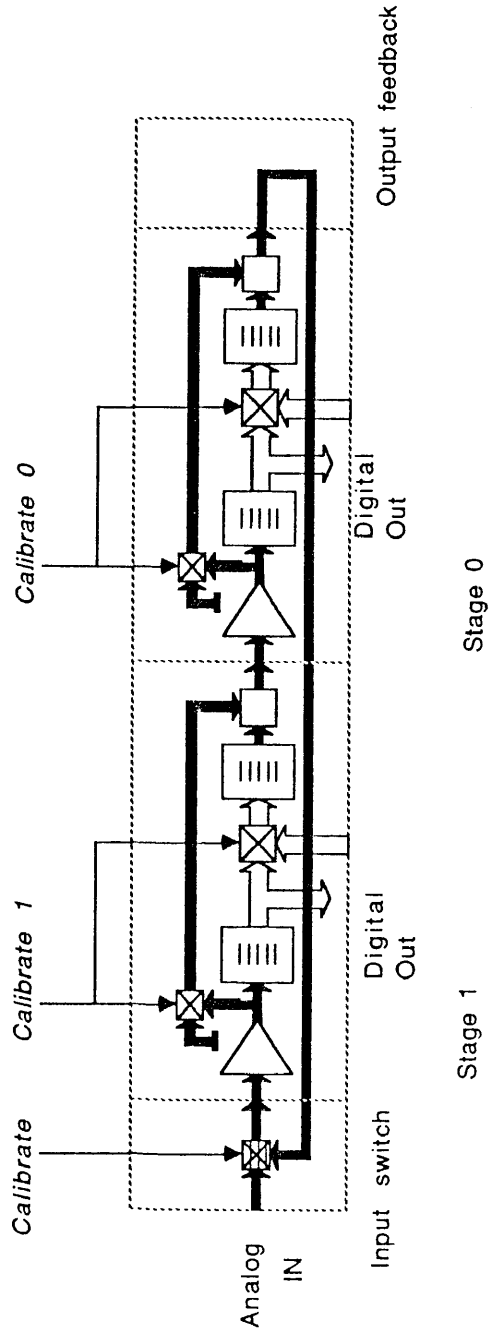


Fig. 79: Reconfigurable Pipelined A/D Converter

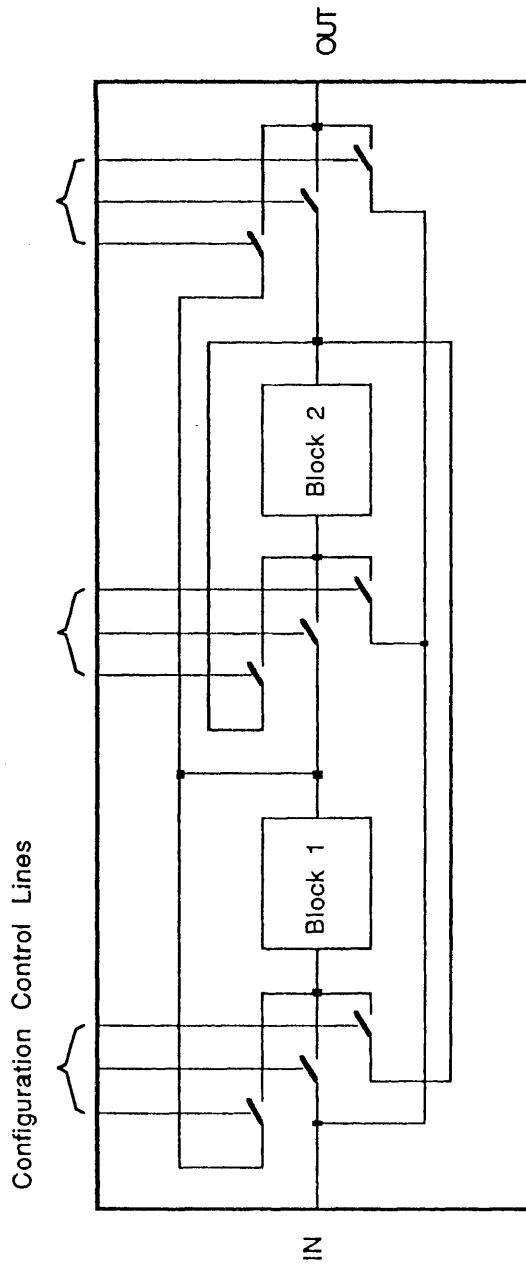


Fig. 80: System with Configuration Control Capability

also be generated externally, e.g. by an external micro-processor. The latter approach has been chosen for the realization of a prototype self-calibrated, pipelined ADC, to be described later.

3. It must be possible to apply control signals to the system, so that the input of some blocks would selectively be connected to one out of a number of reference signals, rather than to the output of another block. Alternatively, these control signals should be able to select one out of a number of possible modes of operation for a block. The purpose is to provide the means to externally excite several degrees of freedom within the block, so that the values of significant internal components could be determined by monitoring the output of the block. The conceptual schematic of one block is shown on figure 81.

This kind of control signals are normally generated by a specific part of the system, further referred to as the block operation controller (BOC). This block operation controller could physically be realized by the same circuit as the configuration controller, either internally or externally.

4. The overall, nominal (desired), transfer function of the system is realized by a particular configuration of blocks. One possible, obvious configuration is such that all blocks are chained together in a linear fashion, with the system input going to the input of the first block in line, and the system output coming from the last one. However, other types of nominal configuration are possible. It is not even necessary that all blocks present within the system be involved in the particular configuration that realizes the nominal transfer function. Some blocks may only be present for the purpose of calibrating the main blocks (e.g. a small recycling converter calibrating the stages of a long pipeline).
5. The transfer function of each individual block can be modified by changing the value of one or more adjustable elements (coefficients) within the block. As a result, the behavior of a block (transfer function or functions, if there is more than one input or output), is normally determined by three classes of elements:
 - fixed components, subject to random variation

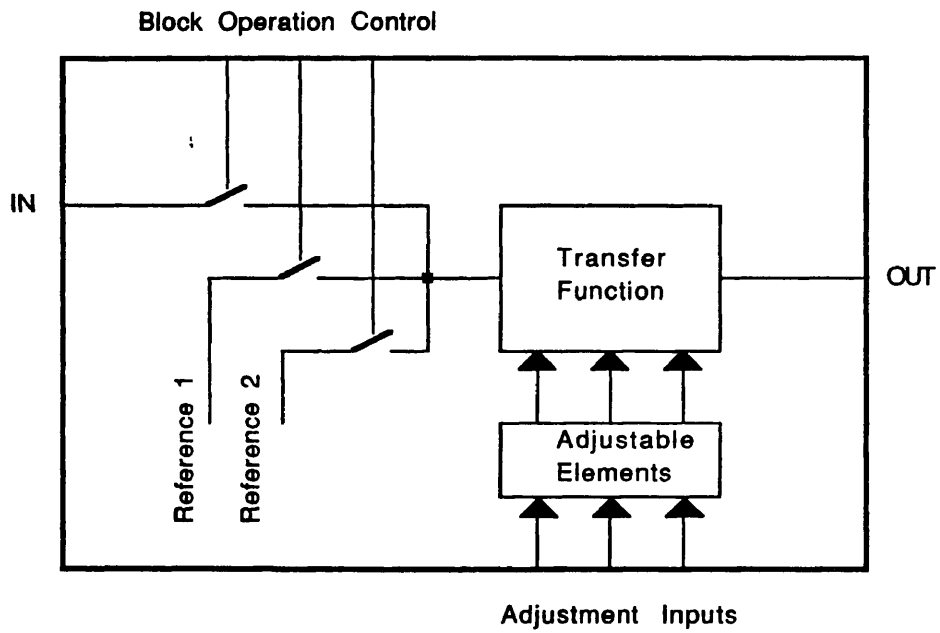


Fig. 81: Conceptual Schematic of One Block

- control signals, selecting a particular mode of operation or a particular input reference level
- adjustable elements

The adjustable elements normally are binary codes stored in a digital memory location (like in the digitally corrected pipelined converter). In the most general definition of a system, they could also be implemented by a voltage temporarily stored on a capacitor, a residual magnetic field stored in a piece of magnetic material etc. Once updated, the value of these elements are normally maintained for relatively long periods of time. "Relatively long", in this context, means at least long enough to cover the time between successive calibrations, if the system is to be recalibrated more than once. This time could be as long as several days, or as short as a few milliseconds, depending on the application.

7.4. Calibration Procedure

Accuracy bootstrapping is performed as follows.

1. Appropriate initial values are assigned to the adjustable elements of the different blocks of the system. These initial values should be determined in such a way that if all other components of the system had their nominal values, the transfer function of the system would be the nominal, desired function.
2. A particular configuration of blocks is realized, so that the input of one block (the block under calibration, or BUC) would be connected to a reference signal, while its output is connected to a sequence of other blocks. These other blocks are used to estimate (measure) the first block.
3. The output of one of the measuring blocks is fed back to the block under calibration, in such a way that the adjustable elements of the block under calibration would be updated by the output of the measuring block. In the case of the pipelined converter, a number of stages would be used as a partial pipelined converter, of which the output would be used to update the digital coefficients of another stage, the "block" under calibration.
4. If needed, another combination of reference levels can be applied to the block under calibration, and other adjustable elements of that same block

can updated, in a way similar to the first one. This, of course depends on the number of degrees of freedom (the number of components subject to variations) within the block.

5. Another configuration is generated, in which another block is updated. The block previously under calibration (and now updated), should now be a measuring block. The new block under calibration is updated, by using the output of one or more of the measuring blocks.
6. The process is repeated a number of times, using different configurations, in which different blocks are being calibrated (updated), using the information from the measuring blocks. This is the first level of iteration of the algorithm.
7. After different blocks have been calibrated, the initial configuration can be generated again, and the first block can be re-calibrated, now using the already updated information within the measuring blocks. The whole sequence of different configurations can be repeated a number of times if needed. This is the second level of iteration.

The successive steps in the algorithm are depicted in figure 82. Figure 83 depicts its application to a three-stage, pipelined converter.

7.5. Convergence of the Procedure

Under certain conditions, which obviously have to be taken into account when the system is initially designed, the process described above will evolve to a situation where the adjustable elements of the different blocks do not significantly change anymore when the process is repeated. In other words, for certain configurations, the iterative system identification algorithm may converge to a certain, accurately predictable equilibrium situation.

In particular cases, it is possible to design the system so that this situation would coincide with all blocks having reached a well-determined transfer function, unaffected by the random variations that are inevitably present on the fixed components of each block. As a matter of fact, such equilibrium condition corresponds to a situation where all adjustable components have been adjusted so as to cancel the effect of the random variations in the fixed components.

Under those conditions, the actual transfer function of each block is well-

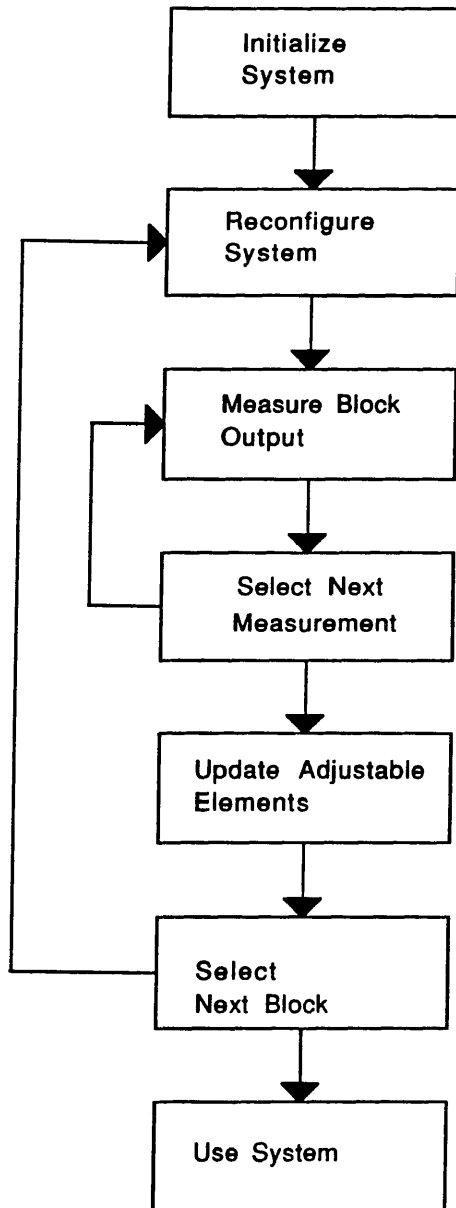


Fig. 82: Accuracy Bootstrapping Algorithm

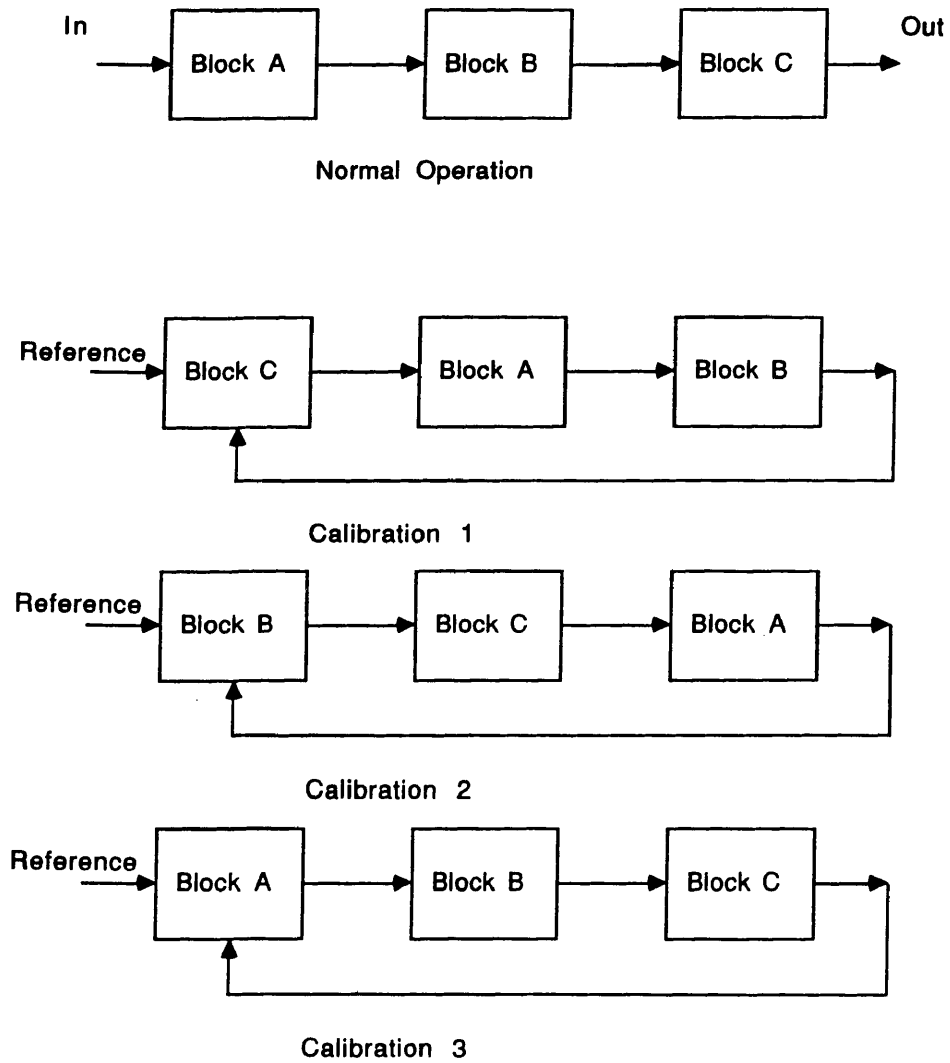


Fig. 83: Application to a 3-Stage Pipelined Converter

defined and the desired overall transfer function of the system can be realized. This is accomplished by configuring the (calibrated) blocks between the input and the output of the system, in a certain, pre-determined, meaningful way.

Unfortunately, establishing a general convergence criterion for accuracy-bootstrapped systems is not an easy task. An attempt at defining such criterion for pipelined ADC's will be made in the next chapter. However, in many cases, the convergence space of the system (the set of allowable errors) so that the whole procedure would still be guaranteed to evolve to one, well-defined state) can adequately be determined by high-level simulations. The variability of the components within the system can be represented by a random number generator. The convergence space can be determined by Monte-Carlo analysis. This approach was taken to determine optimal stage design in pipelined converters, as well as the maximum allowable variation in the components (gains, comparator references, DAC levels etc.).

7.6. Conclusion

In this chapter, accuracy bootstrapping was introduced in general terms, as an iterative technique that may be able to effectively calibrate complex systems that must realize a precise transfer function, but are subject to random component variations. The general architecture of such systems was described. Configuration and block operation controller functions were described. The iterative algorithm was given. However, this chapter stopped short of giving a general convergence criterion for successful calibration. It was mentioned how high-level simulations and Monte-Carlo analysis can often successfully be applied instead.

CHAPTER VIII

AN ACCURACY-BOOTSTRAPPED PIPELINED A/D CONVERTER

8.1. Introduction

In chapter VII, accuracy bootstrapping was introduced as an iterative system identification and calibration technique, from an entirely general point of view. Although no general proof of convergence was given, it was explained how in some cases, an increasingly accurate transfer function can be obtained. One case in which the technique has been observed to work extremely well, is precisely in digitally corrected, pipelined analog to digital converters.

Accuracy bootstrapping appears to be an ideal calibration technique to be used in these applications, since it does not require any precision external hardware. The calibration is performed by the system, in this case the pipeline, itself. In addition, it will be shown that the required overhead circuitry is relatively limited and straight-forward.

Since accuracy bootstrapping does not rely on any external standard, it is obvious that no *absolute* calibration of the system can be obtained. Instead, the technique makes it possible to *linearize* the overall transfer characteristic of pipelined converters to an extremely high level of accuracy. An additional scaling of the digital coefficients, based on the two-point measurement of an external reference (e.g. a band-gap voltage reference) could take care of the absolute calibration, if at all needed.

8.2. Basic Stage

A possible implementation of one pipelined stage is shown in figure 84. This particular stage has two comparators and an interstage gain of -2. It is designed to convert one bit per stage, with $\pm 25\%$ over-and under-range capability (redundancy). Although this arrangement of comparators may seem somewhat particular, a similar structure has also been used in other structures, like the one described in a paper by Ginetti et al. [38]. The authors found it actually happened to implement an RSD (Redundant Signed Digit) division [65].

For this discussion, it is assumed that the signal-carrying variable is a voltage, although the principle can easily be extended to other schemes, e.g. using current,

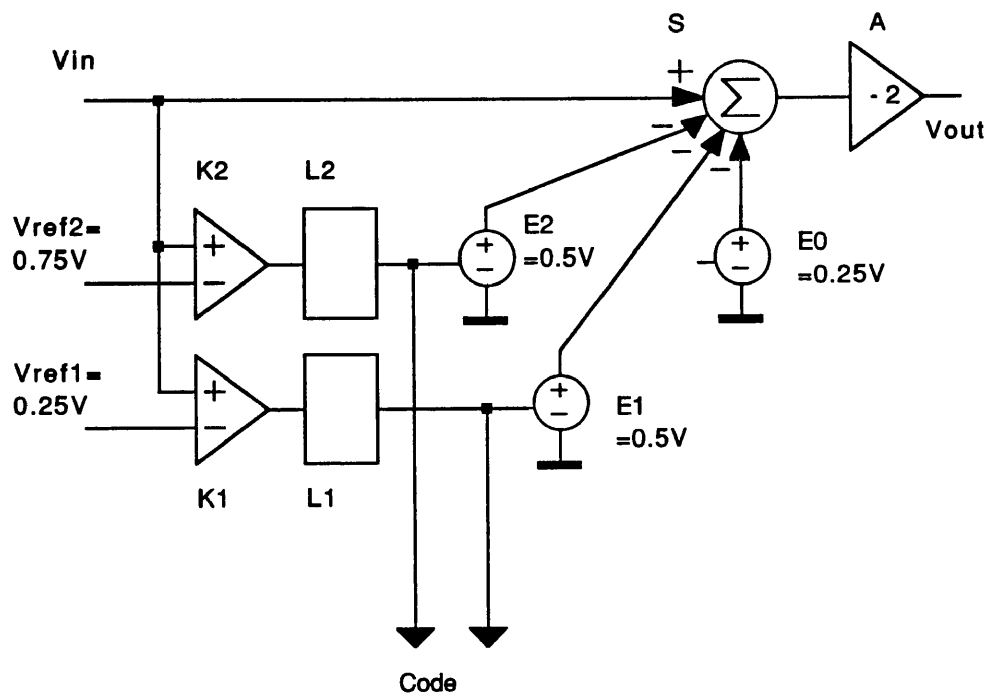


Fig. 84: Schematic of One Pipelined Stage

charge or pulse width coding. A more specific implementation, based on a combination of charge and pulse width coding of the signal, will be discussed in detail in chapter XV.

The gain and the number of comparators of this example have been chosen specifically because this configuration guarantees excellent convergence (evolution towards a very linear overall converter transfer characteristic) of the accuracy bootstrapping algorithm. The actual circuit described in chapter XV is based on the same elementary design of a stage.

The input signal to the stage is a voltage, nominally in the range between 0 and $1V$. The input signal is compared against two reference voltages (nominally equal to $0.75V$ and $0.25V$ respectively), using comparators K_1 and K_2 . The digital outputs of the comparators are latched by the two digital latches L_1 and L_2 . The output of the latches is a two-bit binary code, which depends on the particular value of the input signal. For input voltages below $0.25V$, the code is 00. For voltages between $0.25V$ and $0.75V$, the code is 01. For voltages above $0.75V$, the output is 11 (thermometer coding). Clearly, code 10 cannot occur, unless the comparators have an excessive offset.

Depending on the comparator code, voltages are subtracted from the input signal using the two switchable voltage sources E_1 and E_2 and the summer/subtractor S . A fixed voltage of $0.25V$ is always subtracted from the input signal, regardless of the comparator outputs. This insures consistency between the output range of this stage and the input range of the next stage. The *total* subtracted voltage can be any one out of three values, depending on the comparator code. A code of 00 causes subtraction of $0.25V$. A code of 01 subtraction of $0.75V$ and 11 causes subtraction of $1.25V$.

The summer is followed by a sample/hold amplifier (for a pipelined architecture), with nominal gain of -2. This gain insures that the amplifier output (also an output of the stage) would nominally be in the $0V - 1V$ range, like the input. Actually, the scheme includes $\pm 25\%$ safety margin, since an input voltage of $1.25V$ to the stage will yield an output of $0V$ and an input of $-0.25V$ will yield an output of $1V$, as shown on the nominal transfer curve of figure 85. This margin will allow the stage to operate correctly, even in the presence of fairly large component errors.

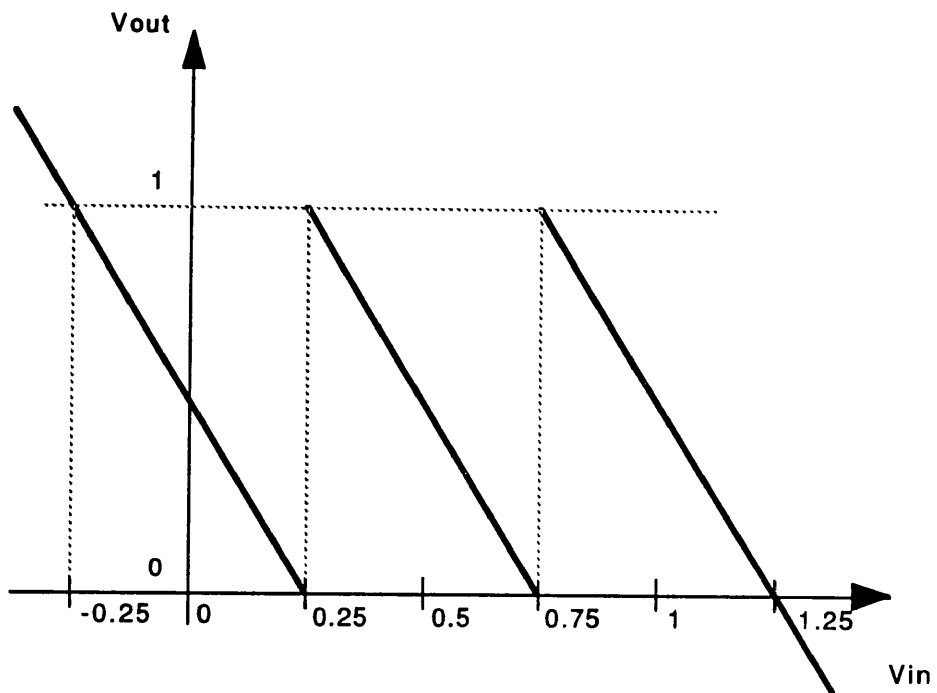


Fig. 85: Ideal Transfer Curve of the Stage (Residue)

In this example, the gain is negative (inverting amplifier). There is no fundamental reason why the gain should be negative rather than positive. It will actually be shown that a positive gain of 2 would also have resulted in successful calibration using accuracy bootstrapping, provided appropriate values were chosen for the voltage sources. However, the negative gain was used in the prototype converter discussed in chapter XV, because of more convenient circuit implementation. There is no fundamental reason for the absolute value of 2 either. A workable scheme could have been built with more comparators per stage and correspondingly higher interstage gain. But the scheme with two comparators provides the largest over-range margin ($\pm 25\%$), and hence the best immunity to component errors.

In addition, when the absolute value of the nominal gain is an exact power of two, the digital calibration hardware can be considerably simplified. We will define the number of bits of each stage as the binary logarithm of (the absolute value of) the nominal inter-stage gain, for reasons that will become apparent throughout this discussion. When the nominal inter-stage gain is a power of two, the number of bits per stage also happens to be an exact integer, in this example 1.

8.3. Component Errors

The main component errors that could affect the transfer function of the stage are:

- incorrect values of the reference voltages ($0.25V$ and $0.75V$ in this case)
- systematic offset of the comparators
- incorrect values of the subtracted voltages (E_1 and E_2)
- incorrect gain of the sample/hold amplifier
- systematic offset of the sample/hold amplifier
- incorrect offset or gain in the summer

It has been shown earlier (chapters V and VI) that all these errors can be lumped into three classes:

- incorrect comparator switching levels (ADC or flash errors)
- incorrect subtracted voltages (DAC errors)
- incorrect interstage gain

On top of these errors, non-linearity errors in the transfer curve of the interstage amplifiers are also possible, as well as random noise or hysteresis introduced by some of the components. We will not focus on any of these effects, since in practice they can be reduced to satisfactory levels using sound analog design techniques. We will simply assume that each inter-stage amplifier has a linear, noiseless, hysteresis-free transfer characteristic, which can be represented by a single (although not necessarily precisely known) value of the gain. Amplifier offset errors are assumed to be included in the DAC level error.

8.4. Pipelined Converter

The operation of one stage can be described mathematically, like was done in a general way in chapter VI. However, in this case, we will initially adopt a slightly different convention regarding the location of the sample/hold amplifier (at the output of the stage rather than at the input).

$$V^{out} = (V^{in} - V^{DAC}(cd)) A \quad (82)$$

With A the actual gain of the amplifier and $V^{DAC}(cd)$ the actual subtracted voltage, determined by the comparator outputs (digital code cd , represented by thermometer coding).

Alternatively, the equation can be rewritten in order to express the input voltage as a function of the code, and the output.

$$V^{in} = V^{DAC}(cd) + \frac{V^{out}}{A} \quad (83)$$

In a complete pipelined ADC, several (L) of these nominally identical stages are cascaded. The previous expression can then be expanded to (the subscript $L - 1$ refers to the first stage, $L - 2$ to the second one etc.):

$$V_{L-1}^{in} = V_{L-1}^{DAC}(cd_{L-1}) + \frac{V_{L-2}^{DAC}(cd_{L-2})}{A_{L-1}} + \frac{V_{L-2}^{out}}{A_{L-1} A_{L-2}} \quad (84)$$

This expression can be further expanded to:

$$V^{in} = V_{L-1}^{DAC}(cd_{L-1}) + \frac{V_{L-2}^{DAC}(cd_{L-2})}{A_{L-1}} + \dots$$

$$+ \frac{V_0^{DAC}(cd_0)}{A_{L-1} A_{L-2} \cdots A_1} + \frac{V_0^{out}}{A_{L-1} A_{L-2} \cdots A_0} \quad (85)$$

For an ideal 16-stage converter with 16 bits resolution and inter-stage gain of exactly -2, the expression could be rewritten as:

$$V^{in} = V_{15}^{DAC}(cd_1) + \frac{V_{14}^{DAC}(cd_{L-2})}{(-2)} + \cdots + \frac{V_0^{DAC}(cd_0)}{(-2)^{15}} + \frac{V_0^{out}}{(-2)^{16}} \quad (86)$$

The last term is the residue of the conversion, which is normally negligible compared to the input voltage range. The output codes of each stage can be used to create a digital conversion result, according to equation (85), which is the DAC relation for this converter (as defined in chapter VI). This is accomplished by using the codes to access one out of three possible memory locations (look-up table). The digital table outputs represent one of the terms of equation (85). This is shown on figure 86. Strictly speaking, the stage now has two inputs (one digital, one analog) and two outputs (also one digital and one analog).

The 16-stage pipelined converter is obtained by combining stages as shown on figure 87. The analog input of the first stage is the overall input of the system. The analog output of the last stage is left open (unused) since it represents the negligible residue. The digital input of the first stage is connected to a pattern of zeros, while the digital output of the last stage represents the conversion result.

One can verify that if all components of the pipeline were accurate (nominal), equation (85) would be implemented by storing the values 0.25, 0.75 and 1.25 in the respective memory locations of the first stage, $0.25/(-2)$, $0.75/(-2)$ and $1.25/(-2)$ in the memory locations of the second stage and $0.25/(-2)^{15}$, $0.75/(-2)^{15}$ and $1.25/(-2)^{15}$ in the last stage. The interaction of the digital outputs of consecutive stages should be such that each one of those values eventually gets added into the conversion result.

Figure 88 shows an architecture that accomplishes the same effect, using nominally identical stages. Using identical stages simplifies the design of the converter, especially when stages need to be reorganized in order to implement the accuracy bootstrapping algorithm.

The digital input of each stage is combined with the memory output of that stage through a digital adder, and then digitally multiplied by a factor -2 (nominal

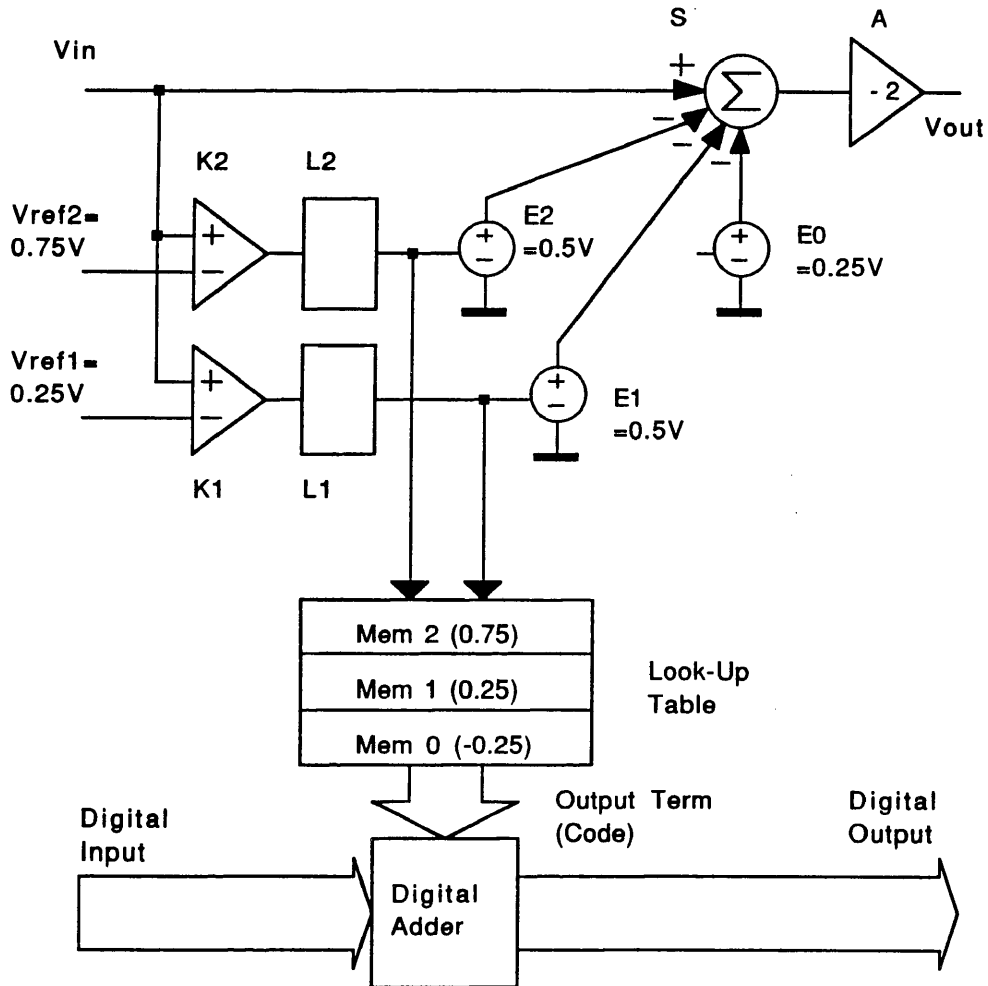


Fig. 86: Converter Stage with Digital Correction

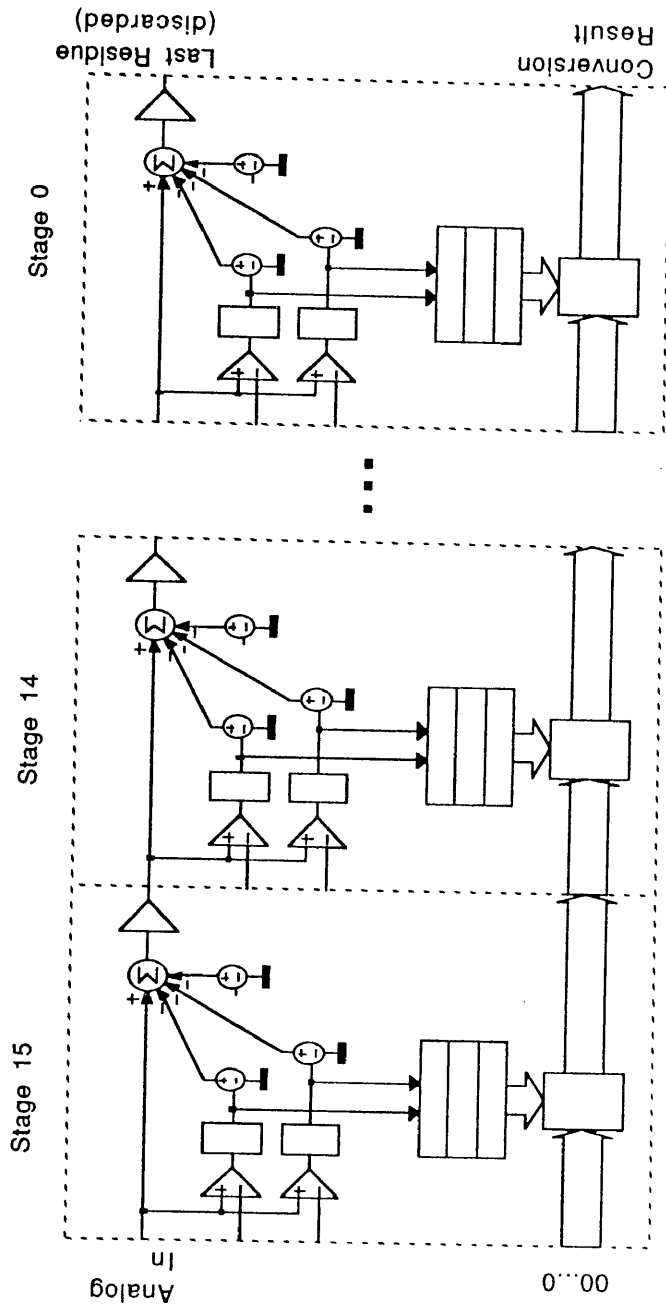


Fig. 87: Schematic of the Pipeline

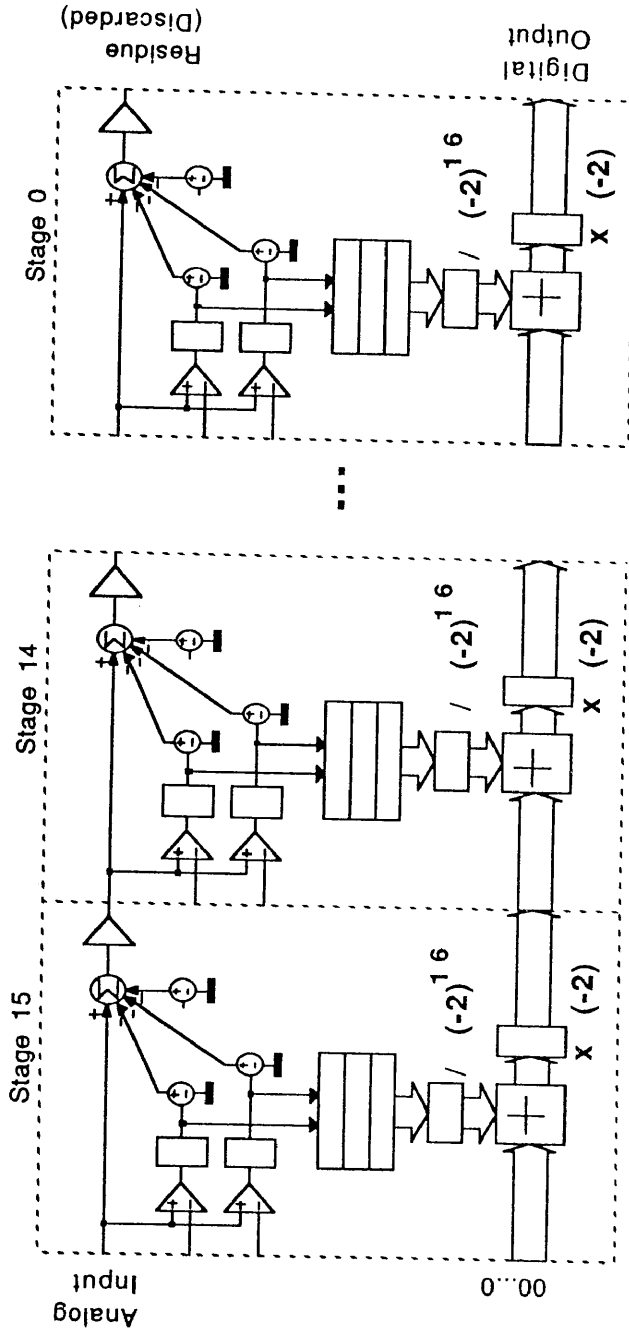


Fig. 88: Pipeline with Error Correction

value of the inter-stage gain). It is clear that the digital input to each stage is the partial sum of equation (85) up to that stage. The digital multiplication by -2 makes it possible to use identical nominal coefficients in the memory locations of each stage. This simplifies the design of the system, since the bits representing each memory location can be laid out in exactly the same way.

The multiplication by -2 does not add excessive hardware. It can be accomplished by inverting the incoming bits (1-complement), adding 1 (2-complement) and shifting all bits one position to the left (multiplication by two). It is clear that using this approach, the initial coefficients in the memory locations of each stage, should be $0.25/(-2)^L$, $0.75/(-2)^L$ and $1.25/(-2)^L$, with L the (fixed) length of the pipeline, in this case 16. For simplicity of calibration, the coefficients can be stored so that their nominal values would be 0.25, 0.75 or 1.25, and a *fixed* division by $(-2)^L$ can be performed by the appropriate arithmetic shift and/or complement operation on the memory output bits.

Unfortunately, due to randomly distributed component errors, slightly different values (coefficients) are required for each stage. These values could be calculated if all actual component values were known, but this is not practical since the actual values of the components cannot be predicted to the required accuracy. In this particular case, accuracy bootstrapping will be able to determine the coefficients to an accuracy of 15 bits or more, even in the presence of several % error on all components.

8.5. Control Hardware

In this application of accuracy bootstrapping, the adjustable components of the pipeline stages will be the digital values, stored in the three memory locations accessed by the comparator output code of each stage. For the sake of reference, we will call these values "coefficients". At the beginning of the calibration cycle, nominal values of 0.25, 0.75 and 1.25 are entered into each stage, to be used as starting values for the algorithm. Since there is a mismatch between the initial values of the terms in the conversion result (equation (85)) thus implemented, and the actual values that *should* eventually be implemented, the pipeline will initially be very inaccurate (6 to 7 bits accuracy is a typical number, instead of the desired 15 bits).

The calibration is performed using a configuration controller, which will successively reconfigure the pipeline, and use some stages to measure and update the coefficients of others. In combination with the configuration controller, a block operation controller will be used to selectively exercise the components that should determine the value each coefficient (in this case, the voltage sources E_1 and E_2 , in combination with the gain stage). Although configuration control and block operation control have been described as two distinct features of accuracy bootstrapping (chapter VI), in practice the required control signals can be generated by a single controller circuit.

In order to make it possible to reconfigure the pipeline, as well as to individually measure the voltage of the sources that are internal to each stage, the basic design of each stage must be slightly modified. This will allow the controller to select the voltage sources externally, rather than through the comparator output codes. It will also allow the controller to disable the analog input to a stage and replace it by a fixed reference level, in this case $0.25V$.

This value is determined as follows. During the calibration, the actual value of each voltage source will have to be measured (nominally, their value is $0.5V$). The measurement will be performed using the other stages of the pipeline. As a result, the voltage source to be measured must be configured in a way that guarantees that the input range of the interstage amplifier following it (nominally $-0.5V$ to $0V$, not including the over-range or redundancy) would not be exceeded.

In order to measure the value of each voltage source, two measurements will be performed: one with that voltage source disabled, and one with the voltage source enabled. When the voltage source is enabled, its voltage is subtracted from the analog input signal through summer S . During calibration, the summer will not be reconfigured in order not to disturb the analog data path; it is desirable to measure the stage in a condition as close as possible to its normal operation condition. For the same reason, the fixed voltage source that subtracts a nominal $0.25V$ from the analog signal will remain active during calibration.

Since $0.25V$ is *always* subtracted from the signal and $0.5V$ will be subtracted only when a voltage source is enabled, the reference level at the input should be $0.25V$. This ensures that the signal at the output of the summer would nominally be between $-0.5V$ and $0V$, as required by the input range of the inter-stage

amplifier.

The $0.25V$ input reference level need not be accurate, since the value of the measured voltage source will be determined as a *difference* of two measurements. In addition, minor variations from the nominal reference level will not cause the input range of the amplifier and subsequent pipeline stages to be exceeded, due to the $\pm 25\%$ over-range capability.

A block diagram of a stage with block operation control capability is shown on figure 89. The analog multiplexer makes it possible to choose between the normal input signal and the $0.25V$ reference level. In practice, this block can easily be implemented using CMOS transmission gates, which can pass or reject an analog voltage level almost without distortion.

The digital multiplexer between the comparator block and the voltage sources makes it possible to control the sources either through the comparator outputs (normal operation, using thermometer coding), or through an externally applied code (during calibration; the code will be 00, 01 or 10). The address multiplexer at the input of the look-up memory makes it possible to access the memory, in order to update the coefficients as the calibration progresses. The line controlling the address multiplexer also controls the read/write mode of the memory.

Figure 89 also shows an arithmetic block, of which the input is the digital value right before the multiplier by -2 . The purpose of this block is to calculate the new coefficients from the measured values, and will be discussed in more detail below.

8.6. Accuracy Bootstrapping Procedure

The calibration starts with the last stage (stage 0). The analog input of this stage is disabled and replaced by the reference level of (approximately) $0.25V$. As mentioned earlier, this allows the switchable voltage sources within the stage to be either enabled or disabled, while the analog signal remains within the range of the interstage amplifier.

The analog output (residue) of stage 0 is connected to the analog input of the first stage (instead of an external input signal). Again, this is accomplished using an analog multiplexer, as illustrated in figure 90. It should be noted that when this multiplexer is in the "feedback" position, all stages of the ADC are effectively

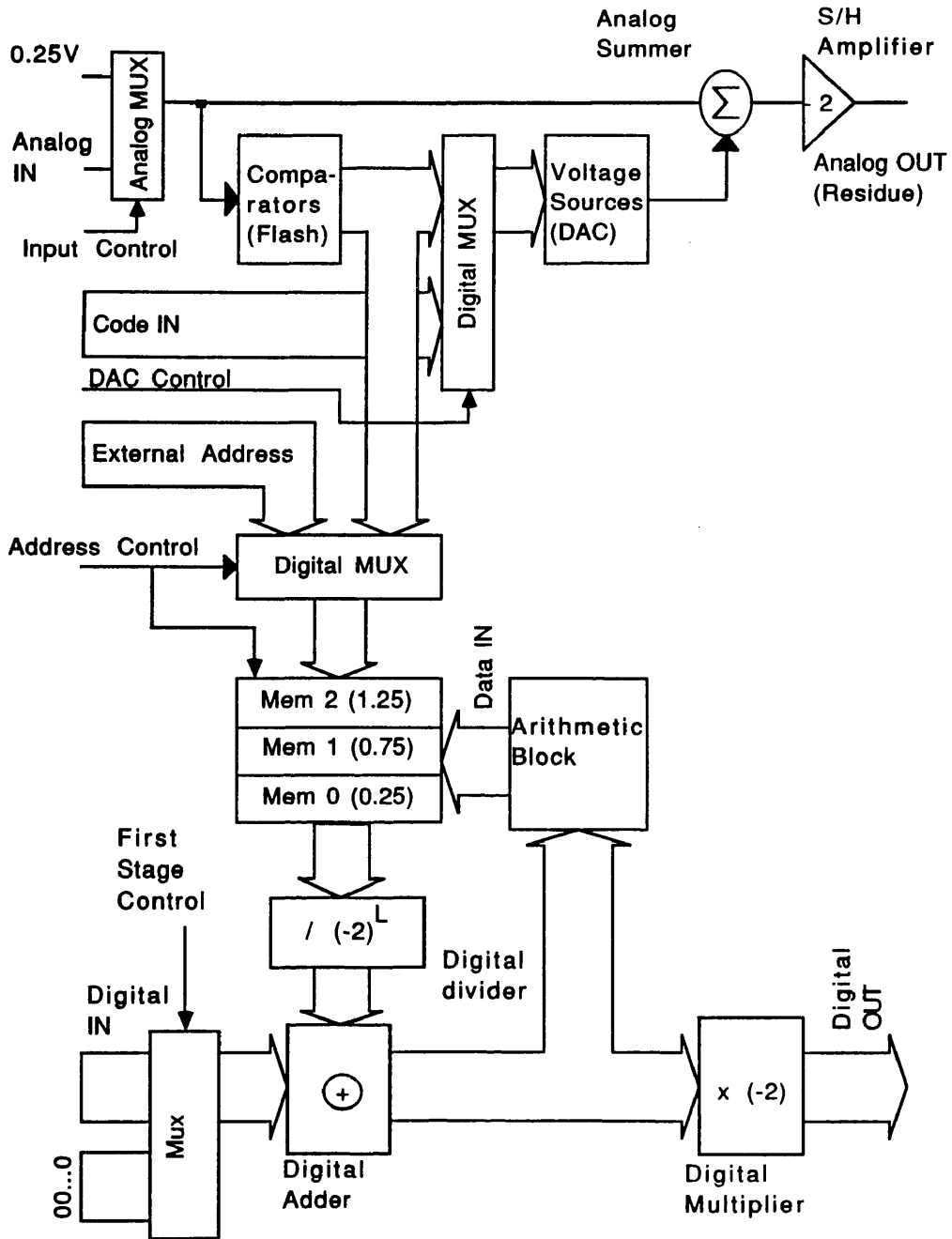


Fig. 89: Calibration Circuitry

connected into a circular (ring) configuration. The ring is only broken by the stage being calibrated, since its input is disabled and replaced by the reference level. In this case (first calibration cycle), the stage being calibrated is stage 0. The digital data paths of each stage are connected into a similar ring, as can be seen on figure 90.

For this step of the algorithm, a temporary 16-stage pipelined converter is formed to measure stage 0. Stage 15 is the first stage, stage 14 the second one etc. The last stage is formed by the first half of stage 0 (the flash section and associated look-up memory). The second half of stage 0 would normally be used to calculate the analog residue. However, since stage 0 is the last stage of the measuring pipeline, that residue can be discarded. Instead, the second half of that stage is used to selectively apply the appropriate voltage source. At the same time, the digital input of stage 15 is grounded rather than connected to the digital output of stage 0, since stage 15 is the first stage of the measuring pipeline.

The first 15 stages (stages $15 \cdots 1$), as well as the comparator outputs of the last stage (stage 0), are now used to measure the reference level applied to stage 0. The resulting conversion result (N_1), is temporarily stored. The nominal value of the analog signal at the input of the interstage amplifier should obviously be $0V$ ($0.25V$ input, minus the fixed $0.25V$). The measured value may differ since it is determined using an uncalibrated pipeline.

A minor problem during calibration, is the following. The reference level applied to the stage under calibration (stage 0) is amplified by the inter-stage amplifier of that stage before being passed on to the measuring pipeline (stages $15 \cdots 0$). This is equivalent to using a pipeline of which the inter-stage amplifiers are located at the input of each stage, rather than at the output, like assumed for the derivation of equation (85). Such convention was actually used in chapter VI, and it is clear that the formula to calculate the overall conversion result must be modified from (85) to:

$$\begin{aligned}
 V^{in} = & \frac{V_{L-1}^{DAC}(cd_{L-1})}{A_{L-1}} + \frac{V_{L-2}^{DAC}(cd_{L-2})}{A_{L-1} A_{L-2}} + \cdots \\
 & + \frac{V_0^{DAC}(cd_0)}{A_{L-1} A_{L-2} \cdots A_0} + \frac{V_0^{out}}{A_{L-1} A_{L-2} \cdots A_0}
 \end{aligned} \tag{87}$$

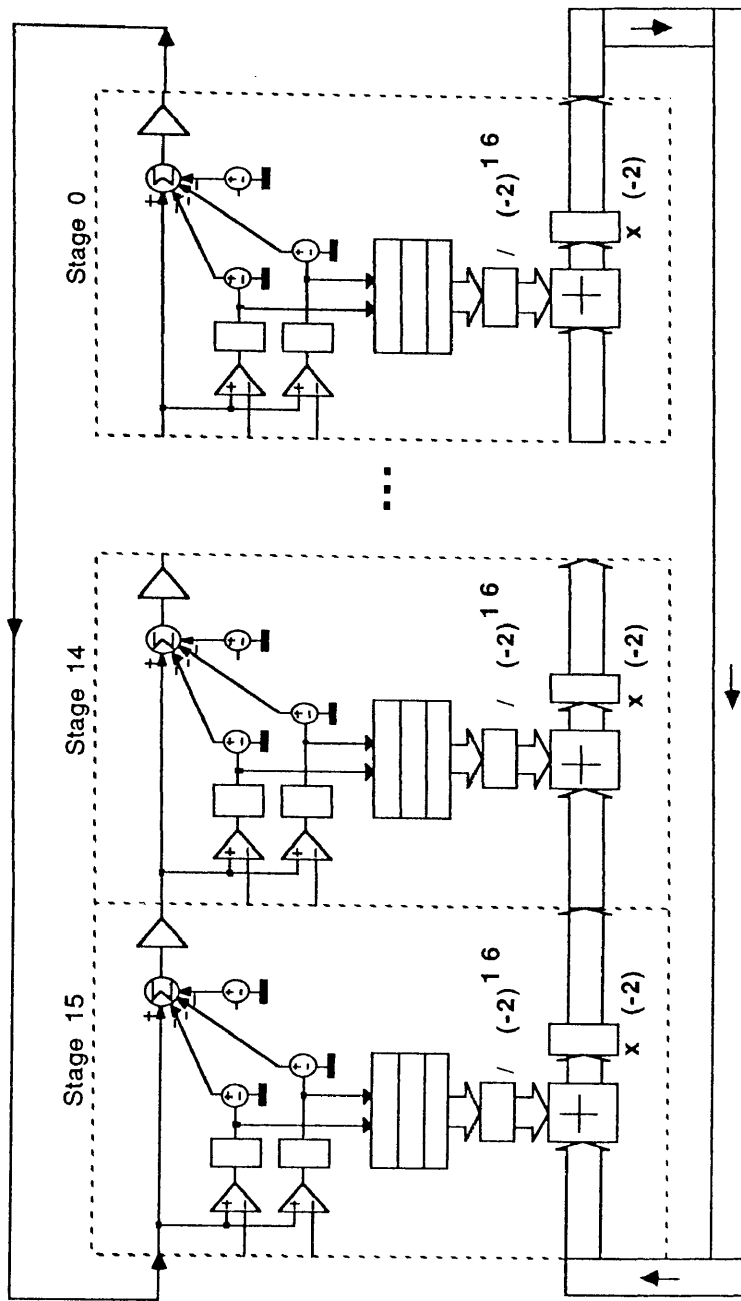


Fig. 90: Feed-Back Configuration

Fig. 90: Feed-Back Configuration

With A_l referring to the gain of the inter-stage amplifier at the *input* of stage l .

One can verify that this distinction will be taken into account if the digital code representing the measurement is taken at the output of the digital adder of stage 0, right before the digital multiplication by -2 is carried out (figure 89), and this after both the analog and the digital signals have proceeded through the complete pipeline (16 stages).

The result of the first measurement, N_1 is stored into the arithmetic block of stage 0 (figure 89). Next, the first voltage source of stage 0 is enabled, while the $0.25V$ reference level is maintained at the input. The first 15 stages and the first half of stage 0 are used again to measure the resulting voltage (nominally $0.25V - 0.25V - 0.5V = -0.5V$ at the input of the first interstage amplifier of the pipeline). The conversion result (N_2) is stored into the arithmetic block.

Next, the same operation is performed, using the second voltage source of stage 0. It should be noted that at this point, the first voltage source is disabled. If it were not, the resulting signal would exceed the range of the interstage amplifier, and no valid measurement could be obtained. The conversion result corresponding to the reference level minus the fixed amount of $0.25V$ and minus the value of the second voltage source is stored. This value is nominally equal to $-0.5V$ (like N_2) and will be called N_3 . This value is also stored.

The three measurements make it possible to estimate the actual values of the two voltage sources. It is clear that the values of the voltage sources represent the two differential DAC levels of the stage, $\Delta V^{DAC}(0)$ and $\Delta V^{DAC}(1)$.

$$\Delta V^{DAC}(0) = V^{DAC}(1) - V^{DAC}(0), \quad \Delta V^{DAC}(1) = V^{DAC}(2) - V^{DAC}(1) \quad (88)$$

$\Delta V^{DAC}(0)$ can clearly be approximated by $N_1 - N_2$. Actually, if the uncalibrated pipeline were accurate and there were no quantization error during the measurement, the two would be exactly equal. For the same reason, $\Delta V^{DAC}(1)$ can be approximated by $N_1 - N_3$. This information is used to update the coefficients stored in the memory locations of stage 0.

First of all, it should be realized that the proposed calibration algorithm does not aim at eliminating overall offset errors in the pipelined converter. As a direct result, the addition or subtraction of a constant error term from each of the

three coefficients within a stage, will not affect overall linearity. This adds one degree of freedom to the determination of the three new coefficients of each stage. We will use this degree of freedom to *define* the value of the first coefficient (the one corresponding to $V^{DAC}(0)$) as 0.25, its nominal value. As a result, the first coefficient will never be updated during calibration.

The middle coefficient however, corresponding to $V^{DAC}(1)$ and originally equal to 0.75, will be replaced by $0.25 + (N_1 - N_3)$. The top coefficient, corresponding to $V^{DAC}(2)$ and originally equal to 1.25, will be replaced by $0.25 + (N_1 - N_2) + (N_1 - N_3)$. The necessary arithmetic operations (simple additions) are assumed to be performed by the arithmetic block of the stage. Updating the top two coefficients concludes the calibration of stage 0.

It should now be clear why the value of the reference signal at the input of the stage is not critical. This value is included in the measurement of N_1 , N_2 and N_3 . Since the differential DAC levels are determined as the difference of two measurements, any systematic error (as well as offset effects during the actual measurement) is cancelled each time. The only restriction is that the combined analog error on all components within the stage, including the reference level, would not exceed the $\pm 25\%$ over-range capability of amplifier and subsequent stages.

After stage 0, stage 1 is calibrated in a similar way. The pipeline formed by stage 0 (which has already been calibrated), stages 15 through 2 and the first half of stage 1 is used during this calibration cycle. The only difference between the calibration of stage 0 and the calibration of stage 1, is that the cyclic structure of pipeline stage is broken in a different place (stage 1).

Next, stage 2 is calibrated, using stage 1, stage 0 (both of which have now been calibrated), stages 15 through 3 and the first half of stage 2. The process is continued until stage 15 (the most significant stage) is calibrated. In this particular set-up, a single iteration through all stages was found to be sufficient in order to calibrate the pipeline to the required level.

The combined linearity error on all terms in equation (85), implemented by the coefficients of the stages after a single calibration pass, will always be below $1/(2^{15})$ on a nominal range of 1 (0V to 1V input). This was determined

through a combination of experiments (Monte Carlo simulations) and theoretical considerations, after compensating all coefficients for systematic gain and offset errors that may remain in the calibrated pipeline (only the *linearity* of the calibrated pipeline is relevant).

This maximum coefficient error translates into an overall integral non-linearity of 15 bits or more (sometimes even in excess of 17 bits). The reason why the accuracy of the coefficients can be *less* than 16 (the resolution of the pipeline), is because of the quantization error, which is present during each measurement and can take any value between -2^{-17} and 2^{-17} (last term of equation (85)).

A tight mathematical proof of the convergence of the iterative calibration algorithm described above, would be rather involved. The beginning of such proof, relying partly on intuition, will be given later in this chapter, but eventually further theoretical research involving the use of statistical methods is suggested.

8.7. Error Tolerance

It was shown how the redundant comparator of the basic ADC stage provided for $\pm 25\%$ over- or under-range capability. The purpose of this redundancy is to be able to absorb the effect of non-nominal components in preceding ADC stages. Such component errors can cause the output from a stage to be outside the nominal $0V - 1V$ analog range.

However, this redundancy does not mean that *any* amount of component errors can be absorbed by the system. An important question that arises, is: "Exactly how large can the individual component errors of a stage be, before even the over-range capability fails?". Fortunately, such analysis is fairly straightforward. It will be derived next for the particular ADC stage of figure 84. The formulas can very easily be generalized for other configurations.

We will assume that the analog input signal to a particular stage is within the range from $-0.25V$ to $1.25V$ (nominal range, including over- and under-range). From figure 85, it is clear that if the ADC stage were nominal, this would still result in an analog output signal between $0V$ and $1V$. In practice, the stage is not ideal, and the output signal may vary outside of the ideal boundaries.

We will assume that three DAC levels of the non-ideal stage are subject to

a maximum absolute (positive or negative) error of ε . The comparator reference levels are subject to a maximum absolute error of θ , and the interstage gain is subject to a maximum relative gain error of ζ . These errors reflect the maximum mismatch between nominal values (0.25, 0.75 and 1.25 for the DAC levels, 0.25 and 0.75 for the comparator references and -2 for the interstage gain) and the actual values of the components.

One can easily verify (figure 84) that the signal at the output of summer S will be subject to a maximum absolute error of $\varepsilon + \theta$, on a nominal range between $-0.5V$ and $0V$. At the output of the stage (after the interstage amplifier), the signal will be subject to a maximum range error ε_R , given by:

$$\varepsilon_R = 2 (\varepsilon + \theta) \zeta \quad (89)$$

In order to guarantee that the over-range mechanism would be effective, the component errors within each stage must satisfy the condition

$$2 (\varepsilon + \theta) \zeta < 25\% \quad (90)$$

8.8. Nominal Code Patterns During Calibration

In order to explain the mechanism by which the calibration algorithm operates to gain increasing accuracy in the pipeline, a good understanding must be gained about the nominal code patterns (comparator outputs) that can occur in the different stages. During calibration, the nominal values applied to the calibrating pipelines are either $0V$ or $-0.5V$. These are the values at the output of summer S (figure 84), right before the interstage amplifier. It should be remembered that during calibration, the pipelines used to measure the DAC levels can be visualized as having an interstage amplifier at the *input* of each stage, and are described by equation (87) rather than equation (85).

When the input voltage to the calibrating pipeline is $0V$ (this is nominally the case when only the reference voltage is applied), the voltage at the output of the first interstage amplifier is nominally $0V$ as well. The corresponding comparator output code of the first stage is 00, and only the fixed voltage of $0.25V$ is subtracted from the analog signal, yielding a residue of $-0.25V$.

This residue is applied to the next interstage amplifier, and the analog signal at the output of that amplifier is nominally $0.5V$. That voltage will generate a comparator output code of 01 in the following stage, which will cause a DAC voltage of $0.75V$ to be subtracted. The residue will again be $-0.25V$, which is passed on to the following stage. It is clear that this pattern will repeat itself over and over, throughout subsequent stages. The conclusion is that the reference level of $0.25V$ will be converted into a sequence of raw codes equal to 00, 01, 01, 01 \dots 01. The corresponding nominal weights (terms) of equation (87) are $0.25/(-2)$, $0.75/4$, $0.75/(-8)$, $0.75/16$, \dots $0.75/65536$. One can verify that their sum is very nearly equal to 0, as expected.

A similar analysis could be made for the case where the reference level is applied, and *one* of the two DAC voltage sources of $0.5V$ (as well as the fixed voltage of $0.25V$) is subtracted. The voltage at the input of the first interstage amplifier will be $0.25V - 0.5V - 0.25V = -0.5V$. This results in a nominal voltage of $1V$ in the next stage, yielding a comparator output code of 11 and a residue of $-0.25V$. Clearly, the pattern of codes corresponding to this situation will be 11, 01, 01, 01 \dots 01. The corresponding nominal weights (terms) of equation (87) are $1.25/(-2)$, $0.75/4$, $0.75/(-8)$, $0.75/16$, \dots $0.75/65536$. Again, one can verify that their sum is very nearly equal to -0.5 , as expected.

The remarkable feature that becomes apparent from this analysis, is that the two sets of codes which would (nominally) occur during calibration, are *identical*, except for the first code (first term of equation (87)). This fact turns out to be essential for the calibration algorithm to work.

8.9. Linearity of the Uncalibrated Pipeline

An important starting point in the error analysis, (and eventually in any reasoning aimed at proving that accuracy bootstrapping never fails calibration of this pipeline), is to derive a mathematical expression for the maximum non-linearity error of an uncalibrated pipeline. A fairly simple expression can be found if the maximum values of DAC and gain errors are known. The expression will be derived for the particular stage at hand, but it should be obvious how the results can be generalized for other configurations.

We will assume again that all the DAC levels of the pipeline are subject to

a maximum absolute error of ε (positive or negative). All gain stages are subject to a maximum relative gain error of ζ . The errors on the comparator levels are not included in this analysis, since it was shown that they do not influence the accuracy of the conversion result (as long as the errors are well within the $\pm 25\%$ range that can be resolved by the redundancy).

Initially, the actual component values are unknown, and nominal values will be used to calculate the different terms in the DAC relation of the pipeline. For the error analysis, we will use equation (87) rather than equation (85), since this is the relation used during calibration. It should be stressed that the nominal input range of the pipelines considered during calibration (with an interstage amplifier at the *input*), is $-0.5V$ to $0V$, and *not* $0V$ to $1V$, like the nominal pipeline. The initial maximum approximation error, ε_i can be calculated as the sum of the maximum error on each term of equation (87).

$$\varepsilon_i \approx \frac{\varepsilon}{2} + \frac{\varepsilon}{2^2} + \cdots + \frac{\varepsilon}{2^{16}} + q + \zeta \frac{1}{2} + 2 \zeta \frac{1}{2^2} + \cdots + 16 \zeta \frac{1}{2^{16}} \quad (91)$$

In this expression, the first set of terms represent the influence of the DAC errors, the term q represents the quantization error. Its maximum value is only about $0.5/2^{16}$, and hence this term will further be neglected. The other set of terms represents the influence of the gain errors. Since the nominal range of DAC voltages for each stage is 1 (1.25-0.25), this value has been divided by the cumulative gain up to the corresponding stage (nominal value of the term), and multiplied by the accumulated relative gain error. Joint terms representing the combined effect of DAC and gain errors have been neglected, since they are significantly smaller than the sum of the other terms.

Equation (91) is overly pessimistic, since it expresses the maximum *absolute* error of the pipeline compared to the nominal case, and does not take into account the fact that overall gain or offset errors are considered irrelevant. The influence of offset errors cannot be isolated from this formula. Any time the errors on the three possible values of one term in equation (87) have the same value, the term is subject to offset only and can be dropped from equation (91). However, the conditions under which this would happen can only be determined from a statistical point of view (involving standard deviations rather than maximum

errors). In a worst-case analysis like this one, possible favorable offset has no effect.

On the other hand, the influence of an overall gain error can be taken into account to a certain extent. If a gain error occurs in the first interstage amplifier of the pipeline, it will have no effect on the overall linearity. This can be concluded from equation (87), since the gain of the first amplifier occurs in the denominator of every term. As a result, the equation for the maximum initial non-linearity error of the pipeline (on an analog input range of $0.5V$) can be rewritten more realistically as:

$$\varepsilon_i \approx \frac{\varepsilon}{2} + \frac{\varepsilon}{2^2} + \cdots + \frac{\varepsilon}{2^{16}} + 2 \zeta \frac{1}{2^2} + 3 \zeta \frac{1}{2^3} + \cdots + 16 \zeta \frac{1}{2^{16}} \approx \varepsilon + 1.5 \zeta \quad (92)$$

8.10. Linearity of the Differential Measurements

During the first step of the calibration procedure, the 16-stage pipeline starting with stage 15 is used to measure the DAC levels of stage 0. The actual measurement is performed differentially, by first applying a reference level (nominally $0.25V$), and then the reference level minus one out of the two possible differential DAC levels (nominally $0.5V$). The resulting measurements have been called N_1 , N_2 and N_3 . The estimates for the differential DAC levels are derived from these values by calculating the differences $N_1 - N_2$ and $N_1 - N_3$.

The nominal sequence of raw comparator output codes corresponding to the measurement of N_1 has been shown to be 11, 01, 01, 01, \cdots 01. In practice, due to the accumulation of component errors, the lower order codes of this sequence may be different, which can be represented as the sequence 11, 01, xx, xx, \cdots xx. (It can be shown that codes before the third stage in the pipeline cannot be changed due to component errors that satisfy the error tolerance criterion set forth above).

Similarly, the nominal sequence of raw comparator output codes corresponding to the measurement of N_2 and N_3 is 00, 01, 01, 01, \cdots 01. Due to the accumulation of component errors, the lower order codes of this sequence may be different, which can be represented as the sequence 00, 01, xx, xx, \cdots xx.

From these considerations, it is clear that when $N_1 - N_2$ and $N_1 - N_3$ are calculated, a number of terms (at least the second term, and usually more) will

cancel each other out. Due to this fact, the differential measurement of the DAC levels may actually be more accurate than predicted by the linearity of the uncalibrated pipeline.

8.11. Mappings and Coefficients

The digital hardware associated with each stage of a pipeline is designed to implement the DAC relation of the converter, as expressed by equations (85) or (87). This is accomplished by the combination of 3-word look-up tables with associated digital multipliers (gain of $1/(-2)^{16}$), as well as the digital adders and interstage digital multipliers (gain of -2). This is shown on figure 88).

As mentioned before, we will refer to the contents of the three memory locations of each stage as the *coefficients* of that stage. It has been shown how the *nominal* values of these coefficients are 0.25, 0.75 and 1.25. The first value is nominally associated with a comparator output code of 00, the second one with a code of 01 and the last one with a code of 11. For the sake of mathematical notation, we will often refer to these codes with the symbol cd , to which we will assign an integer value of 0, 1 or 2 respectively, instead of 00, 01 and 11.

We will refer to the three coefficients of a stage l with the symbols $C_l(0)$, $C_l(1)$ and $C_l(2)$, or in general with $C_l(cd_l)$ or $C_l(cd)$. It is clear that if the pipeline were nominal (no component errors), the coefficients should be equal to the nominal coefficients, referred to as follows:

$$C_l^n(0) = 0.25 , \quad C_l^n(1) = 0.75 , \quad C_l^n(2) = 1.25 \quad (93)$$

The digital hardware implements a mapping between the 16 comparator output codes and the final conversion result. Depending on whether the conversion result is considered before or after multiplication by -2 (figure 88), either equation (87) or equation (85) is implemented. In the remainder of this chapter, we will consider the conversion result *before* the multiplication, since this corresponds to equation (87), which is used during calibration.

The mapping is expressed as a sum of 16 terms, with 3 distinct possible values for each term. These values are determined directly by the coefficients of each stage. For example: the conversion result, CR_{15} , for the pipeline starting

with stage 15 and ending with stage 0, can be written as:

$$CR_{15} = \frac{C_{15}(cd_{15})}{(-2)} + \frac{C_{14}(cd_{14})}{(-2)^2} + \dots + \frac{C_0(cd_0)}{(-2)^{16}} \quad (94)$$

Which clearly implements a mapping between a set of 16 cd values and CR_{15} . The mapping is entirely defined by the $16 \times 3 = 48$ coefficient values associated with the 16 stages of the pipeline.

In a more general way, the mapping implemented by a 16-stage pipeline, with stage L as the first stage, can be expressed mathematically as:

$$CR_L = \sum_{l=L}^{l=L-15} \frac{C_l(cd_l)}{(-2)^{L-l+1}} \quad (95)$$

This formula assumes a cyclic arrangement of the 16 converter stages, such that a value of l with $-16 \leq l < 0$ would refer to the same physical stage as a value of $l + 16$.

The coefficients for the *ideal* mapping (corresponding to a perfectly calibrated pipeline) can be calculated if the *actual* values of all analog components of the pipeline are known. The coefficient values can be derived from the comparison between equation (87) and equation (95). It should be stressed that an ideal mapping does not imply ideal (nominal) analog components, but coefficients that reflect the *actual component values*. The following formula again refers to a cyclic pipeline arrangement of which the first stage is stage L . The interstage amplifiers are assumed to belong to the stage at whose *input* they are located.

$$C_l^{id}(cd) = \frac{V_l^{DAC}(cd) (-2)^{L-l+1}}{\prod_{i=L}^l A_i} \quad (96)$$

8.12. General Linearity Criterion

Equation (92) correctly predicts the worst-case linearity error of an uncalibrated pipeline, under the implicit assumption that all error effects are uncorrelated. This model is realistic for the uncalibrated pipeline only. Once calibration is started, errors due to imperfect knowledge of DAC levels and interstage gains

will not be independent anymore. Actually, it is possible that incorrect gain values used in the terms of equation (87) would be compensated for by corresponding adjustments in the values used for the DAC voltages and vice-versa. In order to perform an error analysis under such circumstances, a more sophisticated linearity criterion is needed.

First of all, we will define perfect *absolute* calibration of a pipeline as the situation in which all terms of the DAC relation implemented by the digital hardware associated with the pipeline would match the terms of equation (87) *exactly*, for *any* possible combination of comparator output codes. This is the case when all the coefficients of the pipeline are equal to the ideal coefficients, defined above (equation (96)).

As opposed to absolute calibration, we will define perfect *relative* calibration as the situation in which all terms of the DAC relation implemented by the digital hardware match the terms of equation (87), except for a possible constant multiplication factor involving all 48 (16×3) values equally, or a constant additive offset equally involving the three possible values of any one term.

The linearity error of the actual pipeline will be defined as the *maximum* difference between any possible sum of 16 values implemented by the digital hardware, after normalization, and the corresponding sum of 16 terms of the ideal mapping. The normalization of the actual terms is defined as the multiplication of the 48 possible values by a constant factor and the addition of a constant offset to each set of 3 possible values for each term, in such fashion as to most closely approximate the ideal mapping.

The wording "most closely" in this definition is to be interpreted as follows. It refers to a linear scaling of the terms of the original mapping so as to *minimize* the *maximum* difference between any sum of 16 values implemented by the actual digital hardware, after normalization, and those implemented by the ideal mapping (minimax criterion).

The problem with this definition is that in order to determine the linearity error of a pipeline, the transformation of its mapping to the mapping most closely matching the ideal mapping (normalization) must first be determined. This is not straight-forward. Instead, we will determine a mapping that almost satisfies

that condition, but not quite. We will call this the reference mapping. Since this is not the most ideal transformation, the result is that the linearity error of the actual pipeline under consideration will be very slightly over-estimated. This is of no concern for a worst-case error analysis; the results will only be slightly more pessimistic than they should be.

The reference mapping is determined as follows. First, the maximum conversion result CR_{max}^{act} that can be implemented by the actual pipeline is determined. This can easily be calculated by adding the largest (maximum) values out of each set of three possible values for each term, which follows from the actual coefficient values of the pipeline (starting with stage L).

$$CR_{max}^{act} = \sum_{l=L}^{l=L-15} \frac{MAX_{cd=0}^2 C_l(cd)}{(-2)^{L-l+1}} \quad (97)$$

Similarly, the minimum possible conversion result CR_{min}^{act} is determined, by adding the smallest (minimum) values out of each set of three.

$$CR_{min}^{act} = \sum_{l=L}^{l=L-15} \frac{MIN_{cd=0}^2 C_l(cd)}{(-2)^{L-l+1}} \quad (98)$$

The difference between CR_{max}^{act} and CR_{min}^{act} defines the digital output range of the actual pipeline, R^{act} .

$$R^{act} = \sum_{l=L}^{l=L-15} \frac{(MAX_{cd=0}^2 C_l(cd) - MIN_{cd=0}^2 C_l(cd))}{(-2)^{L-l+1}} = \sum_{l=L}^{l=L-15} \frac{|C_l(2) - C_l(0)|}{(-2)^{L-l+1}} \quad (99)$$

If all actual component values (DAC levels and interstage gains) of the pipeline under consideration were known, the ideal mapping (absolute calibration) could easily be determined, using equation (87) or (96). For this ideal mapping, the maximum and minimum conversion result (CR_{max}^{id} and CR_{min}^{id}) can be determined in a way similar to CR_{max}^{act} and CR_{min}^{act} , as well as the corresponding digital output range R^{id} .

$$R^{id} = \sum_{l=L}^{l=L-15} \frac{|C_l^{id}(2) - C_l^{id}(0)|}{(-2)^{L-l+1}} \quad (100)$$

The reference mapping of the actual pipeline is determined by multiplying each of the 48 possible terms by the normalization constant $K_N = R^{id}/R^{act}$. This operation equalizes the digital output range of the reference mapping to the digital output range of the ideal mapping. The normalized coefficients C_l^r that would correspond to the reference mapping can easily be derived by multiplying each of the 48 actual coefficients.

$$C_l^r(cd) = C_l(cd) \frac{R^{id}}{R^{act}} \quad (101)$$

Unfortunately, this operation does not yet define the additive constants to be added to each group of three coefficients in order to obtain the best matching between the reference and ideal mapping. To circumvent this difficulty, the maximum linearity error will be determined by comparing the two mappings term by term, on a relative rather than absolute basis.

The maximum linearity error of the mapping implemented by the coefficients of the pipeline is defined as the sum of the maximum discrepancies associated with each of the 16 terms. The maximum discrepancy of a term is defined as the maximum difference between

- the distance between any two different (out of three possible) values of a term of the actual mapping
- the distance between the corresponding two values of a term of the reference mapping.

This definition makes the linearity error totally independent of any systematic scaling or offset in the coefficients of the pipeline, or in the associated terms in the mapping relation. Symbolically, the maximum linearity error ε_{lin} of a pipeline with stage L as the first stage can be written as

$$\varepsilon_{lin} = \sum_{l=L}^{l=L-15} \text{MAX}_{i \neq j} \frac{|(C_l^r(i) - C_l^r(j)) - (C_l^{id}(i) - C_l^{id}(j))|}{2^{L-l+1}} \quad (102)$$

This linearity error can be related to the full analog input range R of the converter (nominally 0.5V for this configuration) in order to determine the relative linearity error, or the number of effective bits N .

$$N = \log_2(R/(2 \varepsilon_{lin})) \quad (103)$$

It should be noted that this definition of effective bits does not include the possible effect of the intrinsic quantization error. As a result, a perfectly calibrated pipeline would theoretically have an infinite number of effective bits. In practice however, this peculiarity is insignificant, since the limiting factor to the eventual accuracy is usually coefficient errors, and not the quantization error. Still, in simulations, pipelines with resolutions of 16 bits were sometimes found to calibrate up to the 17 or 18 bit level.

8.13. Experimental Results

In an attempt to demonstrate the success of accuracy bootstrapping as a calibration algorithm from a practical point of view, a prototype accuracy-bootstrapped 16-stage ADC has been developed. The architecture of the chip is based on the stage presented in this chapter, and will be discussed in detail in chapter XV. Unfortunately, at the time of this writing, experimental results from physical measurements were not yet available.

In the meantime, convergence was established using extensive Monte-Carlo analysis on a digital computer. The operation of the 16 stages was simulated using a high-level behavioral model, programmed in C. A listing of one of the programs is included as appendix A. The influence of random gain errors, offsets, variabilities on DAC levels etc. was simulated using a random number generator, emulating either a constant probability density of the individual errors within a certain range (a few % of the input range), or only the two extremes (maximum positive or negative error).

The particular configuration that was described (nominal interstage gain of -2, two comparators per stage) was never found to fail calibration to the 15 bit level or beyond in *one* iteration involving the 16 stages (one calibration pass, starting from stage 0 and proceeding to stage 15). This was determined after running more than 1000 experiments, with different combinations of errors each time.

These experiments were run so that the maximum combined effect of all errors would not exceed $\pm 25\%$ of the input range for each stage. In other words: the range of all errors was made sufficiently small, so that the analog signal would never exceed the maximum out-of-range capability of the next stage (redundancy). If such conditions were not maintained (e.g. by including errors in excess of 5 or

10 %), failure to calibrate to the full accuracy would be possible, though this has not been observed in every case.

Particular combinations of errors and input signals *can* cause internal signals that would exceed the range of some amplifiers. In some cases, the calibration can recover from the over-range condition and still be fairly successful (e.g. 14 bits final INL). In other cases, recovery does not occur and calibration is completely disturbed (e.g. 8-10 bits final INL).

Obviously, the Monte-Carlo results must be interpreted with caution. The fact that no failure of the calibration was observed for this configuration in this many uncorrelated experiments, determines with almost absolute certainty that the calibration technique works. In other words: the fact that the pipeline calibrates itself cannot be a coincidence.

However, since every possible pattern of errors cannot be systematically simulated, it is not established with absolute certainty that the calibration will *never* fail in the presence of limited component errors. (The simplified model of the 16-stage pipeline is characterized by 16 x 6 individual component errors; if only the extremes were considered, this would still result in 2^{96} possible combinations, which could not each be simulated.)

The same caution would have applied if experimental results from the prototype converter had been available, since a particular prototype only implements *one* single combination of component errors. To make things worse, the exact extent of each possible internal error would have been almost impossible to determine accurately.

These considerations suggest further theoretical work, in order to determine exactly for which subset of the possible error space reliable calibration can be guaranteed under all circumstances. Such complete analysis has not been included in this work, since the mathematical equations tend to become very complex and bulky. However, the foundation for such analysis has been laid in the form of the linearity criterion, and an intuitive explanation for the convergence, which follows below.

8.14. Other Configurations

From the simulations, it is clear that the 16-stage pipeline based on the

stages described in this chapter, exhibits excellent (and in the presence of limited component errors, almost certainly fail-proof) calibration capability. However, one could wonder whether higher resolution pipelines (more than 16 stages) could be calibrated as well. The answer is yes. No fundamental limitation has been observed to the number of identical stages that can be cascaded, while maintaining the possibility to use accuracy bootstrapping to calibrate the resulting pipeline to the limit of its resolution.

Successful calibration to the 23 bit level or beyond has been observed for a large number of experiments performed on a 24-stage pipeline. That pipeline was based on the same elementary stage as was used for the 16 stage example of this chapter. Comparable results were observed for other pipeline lengths, also using a basic stage with two comparators and a gain of -2.

Similarly, successful experiments (no failure to calibrate) were carried out with various pipelines of which the basic stage had two comparators and a positive nominal gain of 2.

Successful calibration has been observed for pipelines with stages that had a different number of comparators, or a different gain. However, these results need a more specific discussion. In all cases, the number of comparators in each stage was equal to the absolute value of the nominal interstage gain, which we will refer to with the symbol G . If the nominal analog input range of a stage was 0 to 1, the comparator reference levels were placed at $1/(2G)$, $3/(2G)$, \dots , $(2G - 1)/(2G)$. (One example is the previously discussed stage, with $G = 2$ and comparator levels at $0.25V$ and $0.75V$, another example is a three-comparator stage with nominal reference levels at $0.166V$, $0.5V$ and $0.834V$).

This particular arrangement of comparator reference levels defines a whole class of pipelined converters. The arrangement appears to be essential for the calibration algorithm to be successful. The program listed as appendix A automatically simulates such configurations, for any value of G , and for positive *or* negative gains. Variations of this program were used for most experiments. The program calculates the linearity of the different reconfigured pipelines during the calibration, according to the worst-case linearity criterion defined earlier.

In each configuration that fits into the general class modeled by the program,

the amount of over-range capability is $\pm 1/(2G)$. This means that the more comparators are included per stage, the less robust the calibration will become, since the error tolerance was determined as

$$2\zeta(\varepsilon + \theta) < \frac{1}{2G} \quad (104)$$

This means that for practical implementations of the algorithm, there will normally be a trade-off between error tolerance and silicon area. Less stages mean less area, otherwise taken up by the analog portion of the stages and the digital correction hardware. However, less stages also means more comparators per stage for a same given resolution, and as a result less error tolerance.

In the first class of converters, it has been observed that the linearity of the pipelines used during successive calibration cycles keeps increasing monotonously, as long as the maximum component errors satisfy the condition set by equation (104). If the cumulative effect of component errors does not satisfy (104), the monotonicity of the calibration may occasionally be disturbed, or the calibration may fail altogether.

Several other classes of pipeline configurations have been simulated. One historically popular configuration (except for the calibration, which is novel) consists of a sequence of stages, each of which has $G+1$ comparators, with reference levels (nominal range 0 to 1) at $0, 1/G, 2/G, \dots, 1$. Such configuration has an over-range capability of $\pm 1/2G$. Unfortunately, successful calibration in such pipelines appears to be occasional rather than consistent. In other words: in some cases, the calibration reaches the required accuracy, but in a significant number of cases it does *not*.

Another major difference between this second class of converters and the first one, is that the linearity of the pipelines used in successive calibration cycles does not increase monotonously. In other words: a few calibration cycles may result in increasing linearity (e.g. from the 7 bit level to the 10 bit level), but the linearity may suddenly fall back to a much lower level (e.g. 8 bits) in the next cycle. This seems to imply that once in a while, a critical interaction of errors can occur, which wipes out the increasing accuracy gained up to that point.

As a result, such converter configurations should be considered worthless for practical or commercial applications. Although we will not give a detailed analysis,

it should be mentioned that the failure to always calibrate has to do with the fact that there are more DAC levels to identify than in the previous case, and that their range (largest DAC level minus smallest one) exceeds the nominal input range of the converter. This can cause unfavorable propagation of errors, which does not occur in the first class of stages.

Successful calibration using accuracy bootstrapping is also possible in configurations with a nominal gain that is not an integer value (e.g. 1.9), and a number of comparators that is *less* than the integer value closest to the value of the gain. (such configuration also exhibits redundancy and error tolerance). This has been observed in a few simulations. However, the number of experiments was not sufficient in order to determine the precise conditions for successful calibration. Further research is suggested in this direction, especially since the use of less comparators (and less coefficient memory) may result in area-efficient architectures.

8.15. Intuitive Explanation of Convergence

Although not a complete mathematical proof, the following intuitive considerations may explain how the calibration process manages to gain increasing overall linearity (as defined by the criterion given above) in pipelines based on the two-comparator stage.

1. The calibration process described above compensates for gain errors in the sample/hold amplifier of each stage.

This is due to the fact that the DAC levels of one stage are measured through the output amplifier of that stage. If the gain of that amplifier exceeds its nominal value by a factor f , the analog output of that stage will consistently be too large, by a factor f . During the measurement of the DAC levels, using subsequent stages, the values of the DAC levels will be overestimated by a factor f .

As a result, the new values stored in the digital memory of the stage are overestimated by a factor f as well. During the next calibration step, the stage that was just updated, will be the first stage in the measuring pipeline. Its stored digital values will implement the first term of equation (87). The other terms are implemented by the following stages. Since all these other terms

contain the gain of the first stage in their denominator, their initial value (which assumes nominal rather than actual gain) will be overestimated by a factor f , thus compensating for the fact that the first term was overestimated.

2. The compensation of errors due to inaccurate comparator switching levels or inaccurate voltage sources can be explained by the fact that the actual value of the two voltage sources of each stage are estimated by forming the difference between two measurements: one of the fixed reference level, and one of the reference level minus one of the source voltages. We previously referred to these differences as $N_1 - N_2$ and $N_1 - N_3$.

It was demonstrated above that the dominant term in each of these expressions was the first one, which reflects the coefficients of the first stage of the measuring pipeline. As a result, the new coefficients of the stage under calibration will be matched fairly accurately to the coefficients of the first stage of the calibrating pipeline. This increases the overall linearity of the next calibrating pipeline, which will use the stage that was just calibrated as the first stage. This process is repeated in subsequent calibration steps. A detailed analysis of the phenomenon is rather lengthy and will be omitted here. However, it could be an important topic for additional research.

8.16. Recycling Converters

It has been shown how accuracy bootstrapping can be applied to pipelined converters, consisting of a number of physically different, but nominally identical stages. Similar principles can be applied to recycling converters, in which the output of one stage is fed back to the input of the same stage for consecutive conversion cycles.

This has been verified using simulations. Success rates in recycling converters even tend to be more favorable than in the corresponding pipelined approaches (at least for the configurations that do not always reach complete calibration). This seems consistent with the intuitive perception that one stage has less unknown components to be identified than a string of stages.

Successful calibration has also been observed in hybrid (partially pipelined) converters, in which the output of a small pipeline (e.g. 2 or 4 stages) is fed back to the input, and recycled a few times before applying another external input

signal.

8.17. Amplifier Non-Linearity

In the given example, accuracy bootstrapping did not attempt to correct for potential non-linearity errors in the gain of the sample/hold amplifiers. This limits the maximum increase in overall accuracy that can be gained in practical applications, since even precision amplifiers always exhibit a finite amount of gain non-linearity.

However, it seems possible to apply a more elaborate scheme of accuracy bootstrapping in pipelined A/D converters, which would correct for gain non-linearity as well. The scheme could use either a piece-wise linear or polynomial approximation for the actual sample/hold gain, as described in chapter VI. However, additional research would be needed in order to properly demonstrate convergence (increasing linearity as the calibration progresses), and in order to determine an optimal hardware organization.

8.18. Practical Implementation

The calibration of a pipelined converter requires the use of a controller, which generates appropriate configuration and block operation control signals. Realization of the different configurations is fairly straight-forward since during calibration, the pipeline is always configured in a cyclic fashion, with the output of the last stage feeding back into the input of the first one. The only thing the controller needs to do, is determine which stage is currently being calibrated. The stage in question breaks the ring formed by the different cascaded stages, since its output is disabled and replaced by the reference voltage.

Updating the coefficients within each stage can be achieved relatively easily for the stages described above (coefficients stored within the stage). However, in some applications, it may be desirable to implement them in an external digital machine, for instance when that machine (e.g. micro-processor) is available anyway as part of a larger framework that uses the A/D converter. The output codes from the comparators of each stage are then considered the outputs of the A/D converter, and the necessary digital circuitry that would otherwise implement the digital memory, adder and multipliers, can be eliminated. The

initial calibration (system identification, determination of the coefficients), as well as the compensation (implementation of equation (85)) can be performed within the external machine.

The controller could theoretically be implemented by that same machine, but this solution is often impractical due to the large number additional external connections (control lines) to the converter. If the converter is implemented as an integrated circuit, each line would normally require a separate bond-pad. Of course, multiplexing schemes could alleviate this problem, but at the price of additional circuit complexity, which might outweigh the complexity of an integrated controller.

8.19. Conclusion

In this chapter, a particular pipelined ADC architecture was described, in which accuracy bootstrapping has been observed to yield dramatic improvements in overall linearity. A detailed discussion of the stages was made. A step-by-step description of the calibration algorithm was given. The experimental methodology used to simulate this and similar architectures was described. In particular, a criterion for the overall non-linearity of the converter was derived. Simulation results and apparent trends and conclusions were discussed. A few additional considerations relating to recycling converters, possible non-linearity compensation and practical implementation of the controller were made.