UNIVERSITY OF SOUTHAMPTON

Testing Techniques and Fault Simulation for Analogue CMOS Integrated Circuits

by

Yavuz Kılıç (BSc, MSc)

A thesis submitted for the degree of

Doctor of Philosophy

Department of Electronics and Computer Science,

University of Southampton

December 2001

UNIVERSITY OF SOUTHAMPTON

ABSTRACT

FACULTY OF ENGINEERING AND APPLIED SCIENCE

DEPARTMENT OF ELECTRONICS AND COMPUTER SCIENCE

Doctor of Philosophy

Testing Techniques and Fault Simulation for Analogue CMOS Integrated Circuits

by Yavuz Kılıç

As size and complexity of Integrated Circuits (ICs) keep increasing, testing those ICs is becoming more challenging task for test engineers. Time-to-Market (TtM) is perhaps the most important parameter in an IC's life cycle. Therefore, one needs to come up with test techniques that give shortest possible TtM, yet cost effective and efficient in terms of acceptable yields. Traditional functional testing is both time consuming and expensive. Alternative technique, structural testing, is well established for digital circuits. For analogue circuits, it seems that it will take a while for structural test to become mature. This is mainly due to the fact that there is still not a standard fault definition for analogue circuits.

This thesis deals with problems related to testing analogue circuits. *Supply current monitoring* is a widely used test technique for digital circuits. Recent research has focused on the application of the technique to analogue circuits. One way to implement the supply current monitoring is to use Built-In Current Sensors (BICSs), which enables Design for Test (DfT) and Built-In Self-Test (BIST). In this thesis a novel BICS is designed for analogue circuits. The BICS was fabricated in 0.8μm AMS CYE CMOS (2.5-5.5V, p-sub, 2-metal, 2-poly). Measurement results done on the fabricated IC confirm the correct functionality of the proposed BICS design.

Marginal voltage screening is another widely used technique for digital circuits. Variable power supply can be used as a technique for the marginal voltage screening. There is some research on the application of the technique to analogue circuits. In this thesis variable supply voltage technique in conjunction with supply current monitoring technique for analogue circuits is further investigated. It has been shown that up to 82% fault coverage for a complex analogue circuit, a PLL (Phase-Locked Loop), can be achieved using this technique.

Fast *fault simulation* is crucial in terms of test generation for both analogue and digital circuits. In this thesis, new methods of speeding-up analogue fault simulation has been proposed. Simulation results carried out on a number of benchmark circuits have shown that employing these techniques along with the analogue concurrent fault simulation can result in up to 100% fault coverage and up to 4.7 times speed-up in terms of the CPU time.

Another way to speed-up the analogue fault simulation is to model an analogue circuit under faulty conditions at a behavioural level. Behavioural fault modelling using analogue HDLs, such as MAST and VHDL-AMS (the IEEE 1076.1 standard) is discussed in this thesis, where it has been shown that using behavioural models developed in this thesis over 373 times speed-up (in terms of the CPU time) is possible compared with the transistor level simulations.

To the memory of my father.

Acknowledgements

I would like to thank my supervisor Dr Mark Zwoliński for his extraordinary guidance and support throughout this thesis and for his invaluable effort finding me all the financial support whenever I needed during my PhD work. To Dr Neil J. Ross for his impartial criticism as my second supervisor and providing me with the measurement equipment I needed for Chapter 3 of this thesis and also with reference letters for my job applications, and Dr Bashir Al-Hashimi for his enthusiastic support and feedback during MPhil transfer examination and afterwards. Thanks also go to Mr Iain McNally for his support on CAD tools used in Chapter 3 of this thesis. I would like to thank all members of Electronics and Systems Design group especially secretaries and Microelectronics group secretaries for providing me with all the facilities I needed during this PhD thesis. I would like to thank to Mr Peter R Wilson for the discussions on behavioural modelling using analogue HDLs and simulations carried out using MAST and SABER, which are covered in Chapter 6 of this thesis. And finally, last but most certainly not the least I would like to thank my family members; my mum, my sister, and my brothers who are the main source of my life.

Contents

LIST OF FIGURES ..... **VI**

LIST OF TABLES ..... **XII**

ABBREVIATIONS ..... **XIII**

1 INTRODUCTION **1**

1.1 Motivation **3**

1.2 Outline of the Thesis **5**

2 LITERATURE REVIEW **9**

2.1 Fault Modelling and Simulation **9**

2.2 Testing **14**

2.2.1 Prototype Test *16*

2.2.2 Production Test *17*

2.2.3 Analogue Test Complexity *19*

2.2.4 Structural Testing *21*

2.2.4.1 Current Based Test Techniques 22

2.2.4.2 Variable Supply Voltage Testing 26

2.3 Design for Test (DfT) **26**

2.3.1 Digital DfT *29*

2.3.2 Analogue DfT *32*

2.4 Built-In Self Test (BIST) **33**

2.4.1 Evaluating BIST *38*

2.4.2 Progress in BIST *39*

2.5 Automatic Test Pattern Generation for Analogue Circuits **41**

3 BUILT-IN CURRENT SENSORS FOR CURRENT BASED TEST **43**

3.1 Introduction **43**

3.2 Built-In Current Sensors **44**

3.3 Shunt Voltage Regulator Based BICS **47**

3.3.1 Simulation and Measurement Results *51*

3.4 Process Variation Independent BICS **53**

3.4.1 RC High Pass Filter *55*

3.4.2 Switched-Capacitor Comparator *57*

3.4.2.1 Non-overlapping clock generation 60

3.4.2.2 CMOS Switch 62

3.4.2.3 Simulation Results for the Comparator 64

3.4.3 State Variable Filter as the CUT *68*

3.4.4 Simulation Results for the BICS *70*

3.4.5 Measurement Results *78*

4 TESTING ANALOGUE CIRCUITS BY SUPPLY VOLTAGE VARIATION AND SUPPLY CURRENT MONITORING **80**

4.1 Introduction **80**

4.2 Marginal Voltage Screening for Digital Circuits **81**

4.3 Variable Supply Voltage Technique for Analogue Circuits **84**

4.4 CMOS PLL for the CUT **85**

4.5 Fault list generation **85**

4.6 Fault Simulation and Fault Coverage **86**

4.7 VDD Change for whole PLL **88**

4.8 Change in VDD of one block **89**

4.9 VSS Change **91**

4.10 Undetected Faults **91**

4.11 Conclusions **93**

5 ANALOGUE FAULT SIMULATION **95**

5.1 Circuit Simulation **95**

5.2 Non-linear DC Analysis **97**

5.3 Transient Analysis **98**

5.4 Logic (or digital) Simulation **101**

5.4.1 Compiled Simulation *101*

5.4.2 Hardware Description Languages *103*

5.5 Mixed-Signal Simulation **105**

5.5.1 Signal Conversion *107*

5.5.2 Synchronisation of Logic and Circuit Simulators *108*

5.5.3 Mixed Simulator Initialisation *111*

5.6 Fault Simulation **112**

5.7 CAFS Hierarchical Fault Simulator **114**

5.7.1 Concurrent Fault Simulation *114*

5.7.2 Fault Simulation with CAFS *115*

5.7.3 Improvements to CAFS *116*

5.7.3.1 Closeness Measurement 117

5.7.3.2 Single Point Closeness Measurement 118

5.7.3.3 Multi Point Closeness Measurement 119

5.7.3.4 Threshold Calculation 121

5.7.3.5 Concurrent Fault Simulation Algorithm 122

5.7.4 Examples *124*

5.7.5 Fault Simulation using Built-in Current Sensor (BICS) *130*

6 BEHAVIOURAL/MACRO MODELLING **133**

6.1 Introduction 133

6.2 Previous Work on Macromodels for Analogue Circuits 135

6.3 Behavioural Modelling 138

6.4 Behavioural modelling using HDLs 148

6.5 Behavioural Fault Modelling using Analogue HDLs 150

6.5.1 Behavioural fault model for the closed-loop inverting opamp in MAST 153

6.5.2 Behavioural fault model for the closed-loop inverting opamp in VHDL-AMS 161

6.6 Conclusion 171

7 CONCLUSIONS 173

7.1 Summary 173

7.2 Original Contributions of This Thesis 176

7.3 Recommendations for Further Work 177

8 APPENDICES 178

8.1 C routines developed for CAFS fault simulator discussed in Chapter 5 178

8.2 VHDL-AMS 184

8.2.1 VHDL-AMS Architecture 185

8.2.2 Differential and Algebraic Equations 186

8.3 Language Elements of VHDL-AMS 186

8.3.1 Overview of VHDL-AMS Models 187

8.3.2 Quantities 187

8.3.3 Simultaneous Statements 189

8.3.4 Conservative Systems 190

8.4 Publications 196

8.4.1 Y. Kılıç and M. Zwolinski, "Testing Analog Circuits by Supply Voltage Variation and Supply Current Monitoring", The 1999 IEEE Custom Integrated Circuits Conference, May 16-19,1999, Town and Country Hotel, San Diego, California, USA. 196

8.4.2 Y.Kılıç, C.D. Chalk and M. Zwolinski, "Design and Realisation of a new Built-In Current Sensor for Mixed-Signal IDDD Test", 5th IEEE International Mixed Signal Testing Workshop, Delta Whistler Resort, Vancouver (Whistler), British Columbia, Canada, 15-18June 1999.....201

8.4.3 Y. Kilic, M. Zwolinski, "Concurrent Transient Fault Simulation of Nonlinear Analogue Circuits", 5th IEEE International Mixed Signal Testing Workshop, Montpellier, France, June 21-23, 2000.....207

8.4.4 Mark Zwoliński and Yavuz Kılıç, "Closeness Measurement in Concurrent Analogue Fault Simulation", ICSES '2000 International Conference on Signals and Electronic Systems, 17 – 20 October 2000 USTROŃ, POLAND.....212

8.4.5 Y. Kılıç and M. Zwolinski, "Process variation independent built-in current sensor for analogue built-in self-test", The Proceedings of 2001 IEEE International Symposium on Circuits and Systems, May 6-9, 2001, Sydney, Australia.....219

8.4.6 Yavuz Kılıç and Mark Zwolinski, "Speed-up Techniques for Fault-based Analogue Fault Simulation", The IEEE European Test Workshop, May 29 – June 01, 2001, Stockholm, Sweden.....224

8.4.7 Yavuz Kılıç and Mark Zwolinski, "BEHAVIOURAL/MACRO MODELLING TO SPEED-UP ANALOGUE FAULT SIMULATION", International Conference on Electrical and Electronics Engineering (ELECO 2001), Bursa, Turkey, 7-11 November 2001.....228

8.4.8 Peter R. Wilson, Yavuz Kılıç, J. Neil Ross, Mark Zwoliński, Andrew D. Brown, "Behavioural Modelling of Operational Amplifier Faults using analogue Hardware Description Languages", 2001 IEEE International Workshop on Behavioral Modeling and Simulation (BMAS 2001).....236

8.4.9 Peter R. Wilson, Yavuz Kılıç, J. Neil Ross, Mark Zwoliński, Andrew D. Brown, "Behavioural Modelling of Operational Amplifier Faults using VHDL-AMS", Design, Automation and Test in Europe (DATE) Conference & Exhibition 2002, Paris, France, 4-8 March 2002, Le Palais des Congrès.....244

9 REFERENCES.....250

LIST OF FIGURES

Figure 1-1. Ideal potential time-to-market reduction due to the use of electronic design automation (EDA) tools for test development [5] 4
Figure 2-1. Simplified general overview of IC production flow [4] 15
Figure 2-2. Prototype test flow [4] 17
Figure 2-3. Production test flow [4] 19
Figure 2-4. Basic configuration used in the technique presented in [43] 24
Figure 2-5. A possible implementation of current test technique suggested in [43] 25
Figure 2-6. A TEM micrograph of silicon dislocation [44] 27
Figure 3-1. Series voltage regulator based BICS [14] 46
Figure 3-2. Shunt voltage regulator based BICS [67] 48
Figure 3-3. Simple 2-stage opamp based CMOS implementation of the shunt voltage regulator based BICS [67] 50
Figure 3-4. Process variation independent BICS design. 54
Figure 3-5. RC High pass filter circuit. 56
Figure 3-6. Transfer curve for the RC circuit given in Figure 3-5. 56
Figure 3-7. Switched-capacitor input offset reduced comparator. 58
Figure 3-8. CMOS realisation of the comparator given in Figure 3-7. 59
Figure 3-9. Non-overlapping clocks. 61
Figure 3-10. Non-overlapping clocks generator [72] 61
Figure 3-11. CMOS Switch. 62
Figure 3-12. Possible distortion on the output of the CMOS switch due to the delay between clocks [70] 63
Figure 3-13. Complementary clock waveforms. 63
Figure 3-14. Equally-delayed complementary clocks generator [70] 64
Figure 3-15. CMOS implementation of the circuit given in Figure 3-14. 64

Figure 3-16. Suitable clock waveforms for the comparator circuit. 65
Figure 3-17. Expanded view of Figure 3-16 66
Figure 3-18. Comparator input and output voltages for TM parameter set. 67
Figure 3-19. Comparator input and output voltages for WS parameter set. 67
Figure 3-20. Comparator input and output voltages for WP parameter set. 68
Figure 3-21. Continuous-time state-variable filter [73] 69
Figure 3-22. The current drawn by the CUT (top) and the monitored current with the BICS for WP parameter set. 71
Figure 3-23. The current drawn by the CUT (top) and the monitored current with the BICS for TM parameter set. 71
Figure 3-24. The current drawn by the CUT (top) and the monitored current with the BICS for WS parameter set. 72
Figure 3-25. Monitored current versus comparator output. 73
Figure 3-26. Equally-delayed non-overlapping clock generator circuit layout used in the proposed BICS circuit 74
Figure 3-27. Switched-capacitor input offset reduced comparator used in the proposed BICS circuit. 74
Figure 3-28. Proposed BICS circuit layout. 75
Figure 3-29. Layout view of the BICS circuit along with the CUT filter. 75
Figure 3-30. Comparator output versus the CUT supply voltage for WS parameter set. 76
Figure 3-31. Comparator output versus the CUT supply voltage for TM parameter set. 77
Figure 3-32. Comparator output versus the CUT supply voltage for WP parameter set. 77
Figure 3-33. Comparator output voltage (top) versus the CUT supply voltage result obtained from the fabricated IC. 79
Figure 3-34. Comparator output voltage (top) versus the CUT supply voltage result obtained from the fabricated IC. 79
igure 4-1 Phased-Locked Loop [79] 85
Figure 4-2. Structural short fault models for NMOS and PMOS transistors. 86
Figure 4-3. Probability distribution function of fault-free and faulty supply currents for calculation of the gap given in (4-1) [81] 87
Figure 4-4 Venn diagram of fault covers of different tests. 90

Figure 4-5 Voltage Controlled Oscillator [79] 93
Figure 5-1. Flow-chart for transient analysis algorithm used in SPICE [4] 100
Figure 5-2. Example logic circuit. 102
Figure 5-3. C description of circuit given in Figure 5-2. 102
Figure 5-4. Simulator synchronisation times [82] 111
Figure 5-5. CMOS Inverter. 125
Figure 5-6. 2-stage CMOS Miller opamp [73] 125
Figure 5-7. Inverting amplifier using opamp of Figure 5-6. 125
Figure 5-8. State-variable active filter [73] 126
Figure 5-9. Leap-Frog Filter [73] 126
Figure 5-10. The faulty current drawn by the CUT (top) and the monitored current with the BICS for TM parameter set for "lpo-1" short fault. 131
Figure 5-11. The faulty current drawn by the CUT (top) and the monitored current with the BICS for TM parameter set for "vin-bpo" short fault. 132
Figure 6-1. 2-stage CMOS Miller opamp used in [107] for behavioural fault modelling. 140
Figure 6-2. Three different configurations used in [107] for the benchmark circuit given in Figure 6-1: (a) Inverting amplifier, (b) non-inverting amplifier, and (c) unity gain buffer. 140
Figure 6-3. Input offset voltage and output voltage versus input DC-sweep voltage for the fault-free inverting amplifier. 141
Figure 6-4. Input offset voltage and output voltage versus input DC-sweep voltage for the fault-free non-inverting amplifier. 141
Figure 6-5. Input offset voltage and output voltage versus input DC-sweep voltage for the fault-free unity gain buffer. 142
Figure 6-6. Input offset voltage and the output voltage for the Type I fault (M4 drain-to-gate short fault for the inverting amplifier configuration). 143
Figure 6-7. Input offset voltage and the output voltage for the Type II fault (M5 drain-to-source short fault for the inverting amplifier configuration). 143
Figure 6-8. Input offset voltage and the output voltage for the Type III fault (M6 open drain fault for the inverting amplifier configuration). 144
Figure 6-9. Input offset voltage and the output voltage for the Type IV fault (M5 drain-to-source short fault for the non-inverting amplifier configuration). 144

Figure 6-10. Macromodel used in [107] to derive the input-output relationship for the closed loop inverting opamp. 146
Figure 6-11. Two stage CMOS opamp used in [35] for behavioural fault modelling. 151
Figure 6-12. The macromodel used in [35] for closed-loop inverting amplifier configuration for the opamp of Figure 6-11. 151
Figure 6-13. Behavioural level DC-offset fault model proposed in [107] 153
Figure 6-14. Behavioural level faulty macromodel for the opamp operating in the inverting amplifier configuration. 154
Figure 6-15. VHDL-AMS behavioural architecture declaration for the behavioural model showed in Figure 6-14. 155
Figure 6-16. VHDL-AMS behavioural architecture declaration for the behavioural opamp model shown in Figure 6-14. 155
Figure 6-17. MAST implementation of m and k for Type III faults. 157
Figure 6-18. MAST implementation of m and k for Type IV faults. 158
Figure 6-19. The input-output relationship found using SABER simulation for the MAST behavioural closed-loop opamp model for the Type I faults. 158
Figure 6-20. The input-output relationship found using SABER simulation for the MAST behavioural closed-loop opamp model for the Type II faults. 159
Figure 6-21. The input-output relationship found using SABER simulation for the MAST behavioural closed-loop opamp model for the Type III faults. 159
Figure 6-22. The input-output relationship found using SABER simulation for the MAST behavioural closed-loop opamp model for the Type IV faults. 160
Figure 6-23. The VHDL-AMS entity implementation of the behavioural fault model derived from Figure 6-10 for the inverting opamp. 162
Figure 6-24. The VHDL-AMS architecture implementation of the behavioural fault model derived from Figure 6-10 for the inverting opamp. 163
Figure 6-25. A VHDL-AMS model of a resistor. 163
Figure 6-26. A VHDL-AMS model of a voltage source 164
Figure 6-27. A VHDL-AMS testbench used with the Hamster simulator to simulate the behavioural model shown in Figure 6-23 and Figure 6-24. 165
Figure 6-28. Slow-transient simulation results using the VHDL-AMS model with Hamster for the positive values of the input voltage source. 166

Figure 6-29. Slow-transient simulation results using the VHDL-AMS model with Hamster for the negative values of the input voltage source. *166*
Figure 6-30. The VHDL-AMS behavioural closed-loop inverting opamp model slow-transient simulation using Hamster for Type I faults for the positive values of vin. *167*
Figure 6-31. The VHDL-AMS behavioural closed-loop inverting opamp model slow-transient simulation using Hamster for Type I faults for the negative values of vin. *167*
Figure 6-32. The VHDL-AMS behavioural closed-loop inverting opamp model slow-transient simulation using Hamster for Type II faults for the positive values of vin. *167*
Figure 6-33. The VHDL-AMS behavioural closed-loop inverting opamp model slow-transient simulation using Hamster for Type II faults for the negative values of vin. *168*
Figure 6-34. The VHDL-AMS behavioural closed-loop inverting opamp model slow-transient simulation using Hamster for Type III faults for the positive values of vin. *168*
Figure 6-35. The VHDL-AMS behavioural closed-loop inverting opamp model slow-transient simulation using Hamster for Type III faults for the negative values of vin. *168*
Figure 6-36. if-then construct implemented in the VHDL-AMS model for Type III faults. *169*
Figure 6-37. if-then construct implemented in the VHDL-AMS model for Type IV faults. *169*
Figure 6-38. The VHDL-AMS behavioural closed-loop non-inverting opamp model slow-transient simulation using Hamster for Type IV faults for the positive values of vin. *170*
Figure 6-39. The VHDL-AMS behavioural closed-loop non-inverting opamp model slow-transient simulation using Hamster for Type IV faults for the negative values of vin. *170*
Figure 8-1. Automatic fault list generation routine in C. *179*
Figure 8-2. A routine to carry out closeness check for the dc analysis. *180*
Figure 8-3. A routine to carry out closeness check for transient analysis. *180*
Figure 8-4. Closeness check routine. *181*
Figure 8-5. A routine to drop faults after the DC analysis. *183*

Figure 8-6. A routine to drop faults during the transient analysis. *184*
Figure 8-7. VHDL-AMS Language Architecture *186*
Figure 8-8. Entity declaration of a signal-flow model *189*
Figure 8-9. (a) Bipolar inverter and (b) its equivalent graph [86] *191*
Figure 8-10. Declaration of nature electrical in package electrical_system [86] . *192*
Figure 8-11. VHDL-AMS model of an ideal diode [86] *194*

LIST OF TABLES

Table 4-I PLL Fault Cover for varying supply voltage 88
Table 4-II PLL Fault Cover for combined tests 89
Table 4-III Cumulative Fault Cover with VDD varied for one block. 90
Table 5-I. Fault simulation results for inverting opamp. 128
Table 5-II. Explanation of abbreviations used in Table 5-I. 128
Table 5-III. Speed-up and Fault coverage for benchmark circuits. 130
Table 6-I. The values of the parameters m and k for different fault types for the closed-loop inverting opamp behavioural model 156
Table 6-II. Comparison of CPU times for different modelling approaches for DC-sweep analysis. 160
Table 6-III. Comparison of CPU times for transistor level transient HSPICE simulations against VHDL-AMS behavioural level Hamster simulations. 171

ABBREVIATIONS

AMS	Austria Mikro Systeme [15]
ATE	Automatic test equipment
ATPG	Automatic test pattern generation
BICS	Built-In Current Sensor
BSIM3v3	Berkeley Short-Channel IGFET Model [17]
CAD	Computer Aided Design
CMOS	Complementary Metal-Oxide Semiconductor
CUT	Circuit under test
CYE	AMS 0.8μm, 2.5-5.5V, p-sub, 2-metal, 2-poly CMOS technology
DAE	Differential and algebraic equation
DFFA	Design for Failure Analysis
DfT	Design for Test
DSP	Digital Signal Processor
EDA	Electronic Design Automation
FET	Field Effect Transistor
HAMSTER	A VHDL-AMS simulator [117]
HDL	Hardware Description Language
HSPICE	Commercial SPICE simulator [16]
IC	Integrated Circuit
IDD	Supply current
IDDD	Dynamic supply current
IDDQ	Quiescent supply current
IFA	Inductive Fault Analysis
IGFET	Insulated-Gate FET

IP	Intellectual Property
JTAG	Joint Access Test Group
LMS	Linear multistep
LTE	Local truncation error
MAST	A proprietary analogue HDL from Analogy [116]
MOS	Metal-Oxide Semiconductor
N-R	Newton-Raphson
R_{drop}	Serial resistance used in shunt voltage regulator based BICS circuit
R-K	Runge-Kutta
RTL	Register Transfer Level
SIP	Silicon Intellectual Property
SISO	Scan-in scan-out
SoC	System-on-a-Chip
TCK	Test clock
TDI	Test data-in
TDO	Test data-out
TMS	Test mode selection
TtM	Time-to-Market
V_{dd}	Positive supply voltage
V_{drop}	voltage across the serial resistance
Verilog	A Hardware Description Language for digital circuits [112]
V_{gs}	gate-source voltage for a MOS transistor
VHDL	VHSIC Hardware Description Language [111]
VHDL-AMS	Analogue and Mixed Signal extensions to VHDL [118] , [119]
VHSIC	Very High Speed Integrated Circuit
V_{out}	Output voltage
V_{ss}	Negative supply voltage

1 INTRODUCTION

Testing of a system is defined in [1] as a process in which the system is exercised and its resulting response is analysed to ascertain whether it behaved correctly. If incorrect behaviour is detected, a second goal of a testing process may be to *diagnose*, or locate, the cause of the misbehaviour. An effect of incorrect operation of a system being tested (or CUT for Circuit Under Test) is referred to as an *observed error.* The causes of the observed errors may be design errors, fabrication errors, fabrication defects, and physical failures [1].

Defects or damage can be introduced during the design, layout, fabrication, or assembly of silicon. Defective Integrated Circuits (ICs) will always be introduced within the product's population. Substantial progress has been made in IC processing and assembly towards reducing the frequency of occurrence of defects, but devices with defects will still exist. This fact establishes the need for IC testing. Testing is conducted to identify faulty components, which are then removed from the otherwise good population. The problem is that the most common methods of testing ICs such as quiescent supply current monitoring (IDDQ) test and Automatic Test Equipment (ATE) cannot adequately discern all the defects that may be present [2].

Testing complex digital ICs has dramatically improved in the last decade. Today, fully automated test solutions are commercially available. Automatic Test

Pattern Generation (ATPG), historically used for low-to-medium complexity designs, is being rapidly augmented with or replaced by Built-In Self-Test (BIST) for today's high-complexity and/or high-performance designs. To implement Design for Test (DfT) efficiently, several key elements must exist: a standard fault model, a standard DfT method, and a standard test access method [3].

In comparison to its digital counterpart, mixed-signal DfT is far behind. For example, there is no standard analogue fault model. Researchers use several fault coverage models to estimate test quality, for example; shorts to other signals, opens, and parametric faults. Also, a standard mixed-signal DfT methodology is not yet available.

Over the past five years, several major approaches have been proposed for mixed-signal test. One approach was based on the function-oriented ATPG that was attempted for digital circuits. This approach failed for digital test because of the great variety in functions and the lack of controllability and observability of nodes in circuits not designed for testability. Since the same problems occur in mixed-signal testing, this proposal is not a viable solution.

Another approach, *virtual test*, includes a set of tools that allow a user to simulate a mixed-signal test program on a mixed-signal tester, including the loadboard and the IC under test. Even though simulation occurs at the behavioural level, simulation times are long. This approach allows test development to start and possibly finish before first silicon. However, it does little to automate test generation, allow test re-use, or provide access to on-chip analogue signals. For some mixed-signal ICs, it may be faster to use the conventional route of manufacturing the IC, debugging it on the actual tester and if necessary re-working the design.

1.1 Motivation

Production test is defined in the literature as the common name for the test steps applied to each manufactured IC at the mass production stage [4]. These steps are discussed in Chapter 2. During production testing, the goal is to distinguish good circuits from faulty ones with minimum cost, where cost is influenced by test time, throughput, and the cost of the test equipment. Unlike with board designs, fault location is not a target because it is not possible to repair or replace faulty components. On the other hand, during the design characterisation, if a circuit has been identified as faulty, it is desirable to find the cause of the failure [5].

Test program development for mixed-signal testers is a major bottleneck in the product delivery cycle for many mixed-signal circuits. This is because unlike digital test development, which is automated with the support of CAD tools for test program generation and verified with the help of software and hardware descriptions of the circuit prior to the availability of silicon, mixed-signal test development is labour-intensive, time consuming, and must be done using fabricated devices and on the tester [5]. Therefore, test engineer has to wait until prototypes of a circuit are available in order to start the test development. The delays due to waiting for prototypes and the lack of automation for mixed-signal test program development greatly increases the product development cycle.

Figure 1-1 shows the idea of using simulation tools in order to reduce Time-to-Market (TtM) of the produced IC. If we could do testing and debugging before the first silicon is available we might be able to greatly reduce the TtM of the produced IC.

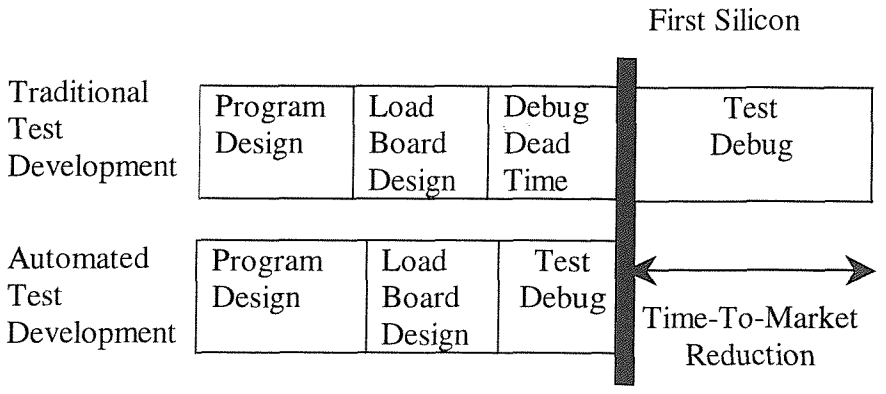


Figure 1-1. Ideal potential time-to-market reduction due to the use of electronic design automation (EDA) tools for test development [5].

During design characterisation if an IC has been found to be faulty it might be useful to diagnose the cause of the failure before it is in high volume production. If faults are identified and located, a circuit can be redesigned to be less sensitive to common failure mechanisms. Therefore we need a means of identifying the component failures. There are two different approaches proposed in the literature for fault diagnosis [5]: *simulation-before-test* and *simulation-after-test.*

Simulation-before-test techniques use a fault dictionary. The faults are then simulated to determine the corresponding responses to predetermined stimuli. Faults are consequently diagnosed by comparing simulated and observed responses. Simulation-after-test techniques, however, begin with failed responses. The failed responses are used to estimate faulty parameter or component values.

Fault diagnosis techniques need fault models. Simulation-before-test techniques are better suited for detecting catastrophic faults and local parametric faults. It is explained in the next chapter what is meant by *catastrophic faults* and *parametric faults.* They might perform less well in detecting global parametric faults, since for such faults the separation between faulty and fault-free responses is less wide. Simulation-after-test techniques, however, are better suited for detecting

problems with global parametric variations and mismatch, and are not well suited for detecting catastrophic faults [5].

The objective of this thesis is mainly to tackle two problems namely, testing and fault simulation for analogue CMOS ICs. A number of testing techniques including supply current monitoring and *marginal voltage* screening have been investigated. It is explained in Chapter 4 what is meant by *marginal voltage*. Fault simulation is key to test pattern generation. As a part of this thesis, analogue fault simulation techniques with regard to simulation-before-test techniques are dealt with in detail in a later chapter. Fault simulation depends very much on the type of the circuit and on the method of the test that will be used after the fault simulation. Analogue fault simulation at transistor level is very slow compared with the digital fault simulation. The reasons why analogue fault simulation is slow and the techniques to speed it up are also discussed in this thesis.

1.2 Outline of the Thesis

The structure of the rest of this report is as follows. In second chapter, a detailed literature review with emphasis on fault simulation and testing issues for analogue/mixed-signal circuits is presented. Supply current monitoring is one way of testing analogue circuits and has proved to be an effective way of doing so [6], [7]-[14]. Two ways of monitoring the supply current of an IC for testing purposes are using ATE or using Built-In Current Sensors (BICSs).

Using BICSs has certain advantages over the usage of ATE, which are explained in Chapter 3 of this thesis, where a detailed investigation of BICSs is given. A new BICS circuit has been developed, which can be used for current based

analogue self-test. The new BICS was designed and fabricated in 0.8μm AMS CYE CMOS (2.5-5.5V, p-sub, 2-metal, 2-poly) [15] technology. The initial simulations for the new BICS design were done using HSPICE simulator [16] with BSIM 3v3 [17] model parameters. The detailed information with regard to this new BICS approach is presented in Chapter 3.

Under normal operating conditions, most of the transistors in an analogue CMOS circuit are in saturation mode. If one can force those transistors to change their normal mode of operation, under a faulty condition one might be able to better identify faulty circuit response from the fault-free one. Therefore, in Chapter 4 a technique to sensitise faults in a complex CMOS analogue circuit is discussed. One way to achieve some control over the behaviour of transistors within an analogue circuit can be varying the supply voltage in conjunction with the inputs. This idea was used by numerous researchers to test mainly small sized analogue circuits with different fault models [18], [19]-[21]. The variable supply voltage as a test technique for analogue circuits investigated in Chapter 4 is based on structural fault models, particularly short circuit faults, where it is aimed to reduce expensive testing cost due to specification and performance based tests by considering testing in a structural manner before the production of the first silicon.

Fault simulation is a very important step to testing. Fault simulation in general, analogue fault simulation in particular is dealt with in Chapter 5. New algorithms in order to speed-up analogue fault simulation have been developed and implemented in C programming language and integrated within a SPICE-like simulator, which has been under development at University of Southampton for over ten years now. The main focus was how to apply the existing techniques of digital domain to analogue domain such that the analogue fault simulation is speeded up.

Therefore, fault-dropping techniques for structural fault-based DC and transient analyses have been thoroughly investigated.

The main difficulty while generating test patterns for analogue and mixed-signal circuits is perhaps the fault simulation. It has been shown that the fault simulation of analogue circuits is at least two orders of magnitude slower than that of similarly sized digital circuits [22] with traditional methods. This is due to the fact that digital circuit simulators use less complex algorithms as explained in Chapter 5 compared with the transistor level simulators, which are used for accurate simulation of analogue circuits. There are a few techniques to speed up analogue fault simulation process namely; fault dropping/collapsing, in which faults that cause similar changes in the circuit response compared with the fault-free circuit response and/or with another faulty circuit response are considered equivalent; behavioural modelling, whereby parts of the circuit are modelled at a more abstract level, therefore reducing the complexity and the simulation time, and lastly new algorithms such as concurrent fault simulation such as concurrent fault simulation [23]-[24] and unified approach for fault simulation [25]. The first technique is dealt with in Chapter 5. Chapter 6 is, therefore, concerned with the second technique, behavioural modelling, in order to speed up the analogue fault simulation. Behavioural modelling for analogue circuits can be done either using macromodels with SPICE-like languages or implementing the mathematical equations that describe the behaviour of the circuit using a high level language such as Hardware Description Languages (HDLs). In Chapter 6 behavioural models for analogue circuits are implemented using analogue HDLs such as MAST and VHDL-AMS.

Chapter 7, first summarises the work carried out in this thesis, and then highlights the original contributions to this thesis, and finally gives some recommendations for further work.

Finally, in Chapter 8 (appendices), the C-code for the algorithms developed in Chapter 5, and a summary of VHDL-AMS language are given. This PhD thesis has resulted in 9 publications, which are also given in appendices in Chapter 8.

2 LITERATURE REVIEW

2.1 Fault Modelling and Simulation

An effective fault model is the one that correctly captures and represents the effect of physical defects on the circuit behaviour. Moreover, it should also lend itself to efficient fault simulation and test generation [26].

The possible defects in a digital circuit can be modelled as, for example, stuck at faults and bridging faults [1]. Therefore, a very large number of possible defects may be reduced to a relatively small number of faults. Defects in analogue circuits, in contrast, cannot be easily modelled by simple fault models. By the very nature of analogue circuits, every defect or even every parametric change might cause a difference in the output waveform or the performance of an analogue circuit. Fault modelling in analogue circuits is, thus, more difficult.

Faults in general are usually classified into two groups: *parametric faults* that degrade the performance of the circuit and *catastrophic faults* that cause the circuit to malfunction. Catastrophic faults are due to local process defects (also known as spot defects) causing shorts, breaks, and device faults (e.g. gate-oxide short), whereas parametric faults are coming from global process defects such as mask misalignment [4], [5], [27], [28].

Due to the increasing importance of analogue and mixed-signal circuits with today's deep sub-micron complex and large SoC (System-on-a-Chip) designs, the problem of testing those circuits will be aggravated. For the economic reasons a functional testing approach is only suitable for small-sized analogue and mixed-signal circuits. General-purpose algorithms for generating real-valued test signals are not practical for the SoC designs, as they get more and more complex in functionality and size [29]. Therefore, research has focused on the alternative approach, structural testing, to reduce the functional testing costs [5].

Historically, test engineers were required to physically modify the CUT (shorting leads, lifting leads, etc.) to analyse the behaviour of the CUT under fault conditions. This can be a time-consuming and costly process. Today, however, test engineers can use simulation as a tool to gain valuable insight into the normal operation of the CUT as well as the operation of the CUT in the presence of faults. By using simulation, the test engineer can obtain the nominal operation and operational range of a device, board, or subsystem. The engineer can also study how CUT would operate if a component were to fail. For each test, the engineer can specify a sequence of single-point, hard faults, analyse the resultant measurement data, and compare the results with previously determined test limits. Analysis of injected faults produces a *fault table* that presents a fault coverage summary of the tests in the proposed Test Program Set (TPS). This table will allow the test engineer to evaluate the quality and fault coverage of TPSs. Using simulation, the test engineer can also analyse the results of TPSs to efficiently isolate a failure in the CUT, leading to potentially significant savings in repair times [30].

Fault simulation requires fault models. The behaviour of a fault model for a given circuit is determined by simulating the circuit model with the injected fault for

a given test stimuli. A fault is then, said to be detected if the performance of the circuit model with the injected fault differs from the performance of the fault-free one in a predetermined manner.

Electronic circuits can be tested by applying a suitable set of test vectors. The smallest number of tests should be applied in order to minimise the time taken to test a device. In general, one test covers more than one fault and each fault may be covered by more than one test. Fault simulation determines the fault coverage of a particular test. *Fault coverage* is a measure of the performance of the applied stimuli. Fault coverage also indicates the faults that are not detected at specific nodes, and therefore provides important information about where potential testability problems may arise. This can lead to modification of the design to improve its testability [31].

As discussed above, fault coverage can be used for comparison between different tests. In general, depending on the definition of a fault and the technique used, fault coverage can be defined as

$$Fault\ Coverage = \frac{total\ number\ of\ detected\ faults}{total\ number\ of\ simulated\ faults} \times 100\% \qquad (2\text{-}1)$$

where total number of simulated faults could be the total number of catastrophic short and/or open circuit structural faults within an analogue circuit, and total number of detected faults could be the total number of faults that are detected among the total number of simulated faults using a supply current monitoring test technique.

In digital fault simulation, a number of copies of the circuit are made, each of which contains exactly one fault, together with the fault-free circuit. In the simplest

case, each faulty circuit is simulated using test vectors as excitations and the result of each simulation is compared with the fault-free simulation result. If a circuit contains n nodes, there are 2n possible stuck at faults, namely each node either stuck at 1 or at 0, and therefore 2n+1 simulations are required in total [32]. As n gets larger, the fault simulation of 2n+1 circuits will become very expensive in terms of CPU time. Consequently, a number of fault simulation techniques for digital circuits, such as parallel fault simulation and concurrent fault simulation, have been developed to reduce the simulation time [1].

There may be cases that some parts of a faulty circuit will give the same response as the fault-free circuit. Concurrent fault simulation technique uses that benefit in such a way that to reduce the simulation time for digital circuits. In concurrent fault simulation, the simulation of n faulty circuits and the fault-free circuit are carried out at the same time, while comparing each faulty circuit response with the fault-free circuit response. The differences between each faulty simulation and the fault-free simulation are then evaluated in order to avoid unnecessary computation [1], [23].

In contrast, the fact that even fault-free simulation of analogue circuits can take orders of magnitude longer than the simulation of similarly sized digital circuits means that the fault simulation of analogue circuits and hence the derivation of the test stimuli can be prohibitively expensive [23]. Recently there has been some research towards investigating algorithmic methods for the efficient fault simulation of analogue circuits [23]-[25]. The application of concurrent fault simulation idea to analogue circuits was first suggested in [23]. By simulating a number of faulty versions of a circuit concurrently with the fault-free version, those parts of a faulty circuit that behave in the same way as the fault-free circuit at any instant in time

would not need to be re-evaluated, thus potentially saving computational effort. Analogue fault simulation problem is investigated in more detail in Chapter 5.

The use of behavioural simulation in the design of large printed circuit boards and systems is well known [33]. Using behavioural models of complex or large circuitry, designers can simulate and analyse their systems in a reasonable time. While a single simulation of such systems at the primitive, or element, level may take hours, or even days, a behavioural simulation may take only minutes and still provide sufficient detail of the system's operation. These same techniques can be applied to fault simulation of large or complex CUTs. Simulating faults for the purpose of developing an efficient sequence of tests to be run on the ATE involves multiple simulation runs and potentially long simulation times due to the insertion of the faults. Using behavioural modelling approach, one can keep fault simulation times reasonable for large systems. The most critical task in creating the behavioural model of a large system is perhaps to decide which effects need to be captured to provide an accurate model of system operation and still provide enough detail to allow required analyses [33]. In Chapter 6 of this thesis, behavioural modelling using HDLs for analogue circuits is studied in more detail.

Behavioural modelling, in general, can be done in two ways: using macromodels that are implemented with fewer components, such as controlled sources as used within SPICE-like languages, than the actual circuit, or implementing the mathematical equations that describe the circuit using a high level language such as HDLs or C programming language [35]. Developing macromodels for analogue circuits, particularly for faulty circuit behaviour, is not a trivial task [34]-[38]. Therefore, recent efforts are towards the use of HDLs in behavioural

modelling approach, as it is much easier to do behavioural modelling using HDLs compared with SPICE-like macromodel approach [35].

2.2 Testing

The advent of IC technology and the scaling of transistor sizes have allowed the development of much larger electronic systems such as today's complex SoC designs. For most of its history, the semiconductor industry has most visibly appeared to focus on digital technology due to its predictable scalability [39]. Digital design techniques have become predominant because of their reliability and low power consumption. However, although large electronic systems can be constructed almost entirely with digital techniques, many systems still have analogue components. This is because signals coming from storage media, transmission media, and physical sensors are often fundamentally analogue. Moreover, digital systems may have to output analogue signals to actuators, displays, and transmission media. Clearly, the need for analogue interface functions like filters, data converters (analogue to digital and digital to analogue), phase locked loops, etc., is inherent in such systems. The design of these interface functions as ICs has reduced their size and cost, but in turn, for testing purposes, access to nodes is limited to primary inputs and outputs, making it more difficult to locate component failures when circuit specifications are not satisfied [5].

Functional (specification-based) testing of analogue and mixed-signal circuits is expensive due to the large number of circuit specifications to be tested. Therefore, structural testing has been introduced as a solution for cost-effective test generation. Structural test generation starts with a list of faults (fault list) that model the faulty

conditions of the CUT. Then, a test set is generated to detect the modelled faults [40].

From the specifications to the finished IC, in general testing can mainly be classified in two different categories; prototype testing and production testing [4]. The overview of IC design and production can be simplified as given in Figure 2-1.

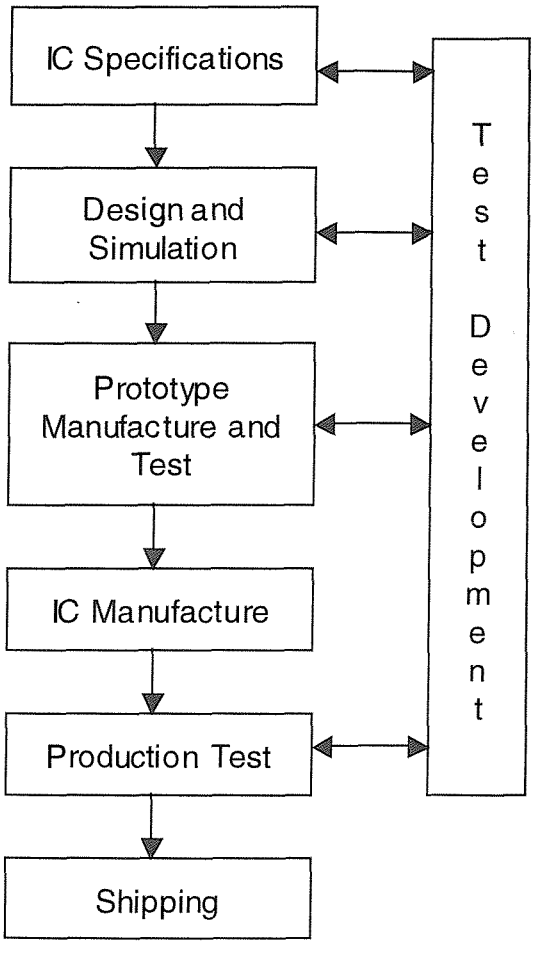


Figure 2-1. Simplified general overview of IC production flow [4].

As can be seen from Figure 2-1, test development is in interaction with many steps during the IC production. Therefore, testing is not just a step in IC production cycle; rather it is involved in almost every step of the product.

Various production tests, such as functional, parametric, probe and final testing all cover different sorts of failures. The aim of the production test is to determine in an economically viable way whether the product satisfies all the requirements concerning functionality, performance, quality, and reliability [18].

2.2.1 Prototype Test

Prototype test is mainly concerned with the IC characterisation phase [4]. Therefore, instead of a pass/fail decision, the prototype test results in a set of performance specifications for the IC. Prototype test consists of two steps; design debug and design evaluation [4]. Prototype test flow is depicted in Figure 2-2.

Design debug is the first and most informal test that an IC undergoes. The designer verifies the correct functionality of the IC through the use of measurement equipment. At this stage if the IC is not functioning as expected, the designer then debugs the design and carries out required modifications to the design. If the IC passes this stage, however, the design is evaluated by applying full functional tests and measuring the specified parameters [4], [30].

As the prototype testing is performed only on a small number of manufactured ICs, the test time is not a primary limitation. The test choice and measurement accuracy on the other hand are important since a full evaluation is required. Moreover undetected errors due to the design can cause delays in the whole product development cycle [4].

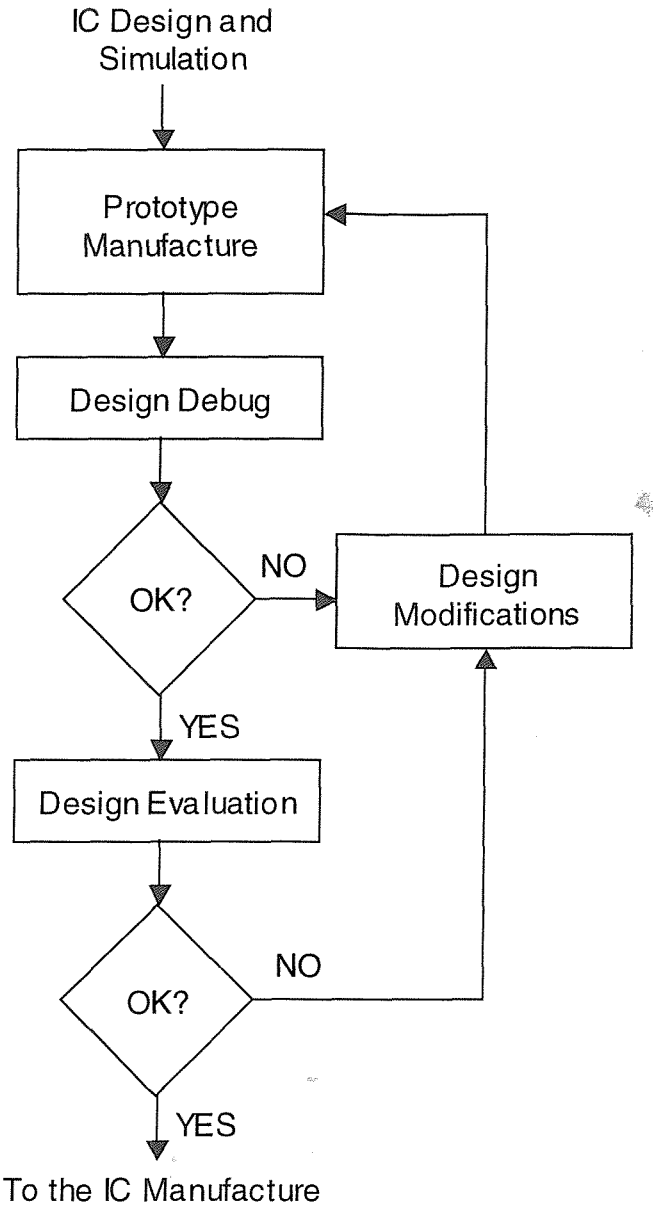


Figure 2-2. Prototype test flow [4].

2.2.2 Production Test

Production test can be defined as the common name for the test steps applied to each manufactured IC at the mass production stage [4], [5]. These steps can be applied both before and after the IC is packaged. Hence, they are called *wafer* test and final test, respectively [4]. The word *wafer* refers to the basic physical unit,

which generally contains a large number of ICs, used in IC processing. Production test flow is given in Figure 2-3.

The wafer test is generally the application of and measurement of DC and low-frequency AC signals to the IC under test. These are mainly general functional and parametric tests in which the connections of the power lines are checked. It is not often possible to apply high frequency tests or tests that require very accurate timing measurements due to the insufficient controllability and accessibility to the wafer probe pins. The latter mentioned tests are applied at the final test stage once the IC is packaged [4].

Wafer test is used to capture the defective chips. The chips that pass the wafer test are then packaged and the final test is applied. Final test involves checking the bonding connections, application of the digital test patterns (if applicable) and measurement of main analogue specifications [4].

Since the test development for the analogue circuits has been mainly specification driven it cannot guarantee certain fault coverage. Moreover, testing the performance of state-of-the-art analogue circuits may require the application of high performance stimuli.

In the next section some of the causes that make analogue functional testing more complex than its digital counterpart are discussed.

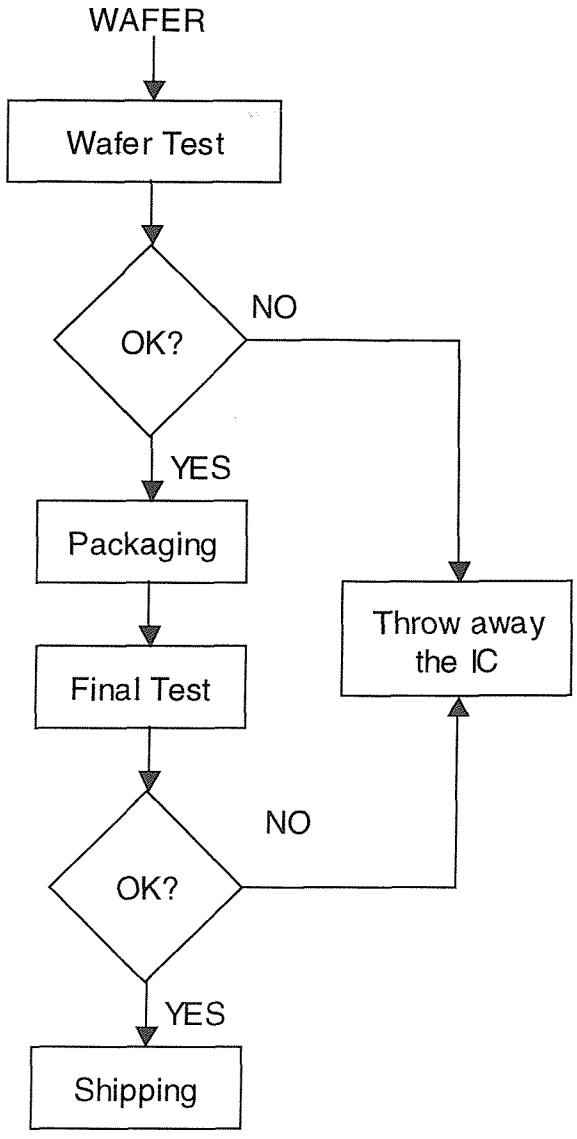


Figure 2-3. Production test flow [4].

2.2.3 Analogue Test Complexity

The causes of the analogue test complexity have been recently addressed in [41]. These are:

- Unlike digital circuits, analogue circuits do not have the binary nature. The time and the voltage continuous nature of analogue circuits make them further susceptible to defects. Therefore, more test procedures are

needed to discriminate between various faulty conditions and the non-faulty condition.

- Analogue systems are often non-linear; thus their performance depends heavily on circuit parameters. Process variations within allowable limits can also cause unacceptable performance degradation. The deterministic methods of such systems are often inefficient.
- In digital circuits, the relationship between input and output signals is Boolean in nature. Many digital DfT schemes simplify this relationship to reduce the test complexity. On the other hand the input-output relationship in analogue circuits is non-Boolean, complex and difficult to model.
- Digital DfT schemes, based on some kind of a structural division of the circuit when applied in analogue domain, are also largely unsuccessful because of their impact on the circuit performance.
- In the digital domain, there exist a range of fault models such as bridging faults, delay faults, stuck-at faults etc. These models or abstractions form the basis of representing the faulty circuit behaviour as well as the test pattern generation. In the analogue domain the effectiveness of these models is questionable. Moreover, in the absence of an acceptable fault model, test generation has been ad-hoc and testing has been largely functional (specification oriented) in nature.
- Since different specifications are tested in different manners, it makes analogue functional testing costly and time consuming. Moreover, extra hardware is often needed to test various specifications.

- Limited functional verification does not ensure that the circuit is defect-free and escaped defects pose quality and reliability problems.

As stated above, analogue test in a functional manner is not satisfactory in terms of TtM and economical reasons. Therefore, the next section discusses an alternative way of testing.

2.2.4 Structural Testing

For analogue circuits, test development is still very much based on functional and performance specifications. A test developed on the basis of the functional and performance specifications can neither give a guarantee with respect to certain defect coverage nor the quality and reliability level to be expected. The device can only be assured to operate correctly under the tested conditions.

This has resulted in the question; whether alternative test techniques can be used which on the one hand guarantee certain defect coverage but on the other hand do not require high performance tests.

Comprehensive specification testing of analogue circuits is costly both in terms of time and in test equipment. Moreover a specification test will not necessarily detect all the defects that could occur during manufacture. Whether or not these defects compromise the functionality of the circuit, they could reduce reliability. Research has started focusing on a structural fault-based strategy to overcome these difficulties.

With a test technique using a structural fault model, test sets are designed to target a specific set of modelled faults. This means that the quality of any set of potential test vectors can be easily quantified in terms of the fault coverage they

provide. Structural testing, however, suffers from establishing a satisfactory formal link between the fault detection and the satisfaction of design specifications [28].

In structural testing, DC, AC, and transient monitoring of the output voltage or supply current can be used as a means of testing [32]. As a part of this thesis, current based test techniques using built-in current sensors, and variable supply voltage testing as a way of structural testing are investigated in more detail in Chapter 3 and Chapter 4, respectively.

2.2.4.1 Current Based Test Techniques

One way of implementing current based test technique, DC supply current monitoring (also known as quiescent supply current monitoring or IDDQ test), is well known for digital circuits [42]. The supply current drawn by a CMOS IC during normal operation consists of two parts, namely, dynamic and static current. The dynamic current test is not practical for digital ICs since further processing will require the measuring equipment that must have a sampling frequency greater than the highest frequency seen in the supply current response, which could be several times that of the IC's clock frequency. On the other hand, measuring the static current can be achieved at much lower frequencies [32].

The supply current in MOS and bipolar analogue circuits has a relatively high quiescent value [6] in which case faults can be masked [32] because the difference between faulty and fault-free responses is relatively small. Therefore, DC supply current monitoring can be misleading for analogue circuits. In order to overcome this, some researchers have focused on measuring the RMS value of the supply current variation for analogue circuits [6], [32]. This removes the quiescent component, potentially avoiding the masking of faults, which might give better fault coverage [32].

Supply current monitoring can be implemented either using ATE or BICS. Using a BICS is more advantageous in terms of: test equipment costs; increasing the testing rate; improving the fault detectability and observability of the CUT; higher current sensing resolution and avoiding the influence of I/O currents which may dominate the chip's total current [7]. Detailed investigation of using BICS for supply current monitoring is discussed in Chapter 3.

Another practical way of using current measuring technique for analogue circuits is to monitor each current branch separately. A test method for analogue part of ICs was proposed in [43] that claims to determine whether an IC is good or not by measuring the currents flowing through its constituent circuits. The proposed technique was mainly targeted to detect defective circuits during the wafer testing. It was claimed that one could test both static and dynamic current through the use of the technique proposed in [43].

One way to measure the current flowing into a circuit in an IC is to measure the voltage drop on the interconnect in either the ground or the supply rail. This voltage drop is usually at the range of millivolts (mV) [43]. Practically, the performance of the IC will decrease as this voltage drop increases. Measuring a few mV in a test factory, however, is not very easy task due to the high levels of electromagnetic interference (EMI) [43]. Therefore, in order to do the measurements reliably the signal should be amplified.

In [43], the authors suggest to use a differential transistor pair in order to amplify the few mV voltage drop mentioned above. A differential pair of bipolar transistors is shown to give a 4% change of the collector currents for 1 mV input voltage change [43]. There will always be an inaccuracy with this technique due to the inherent offset voltage coming from the non-ideal behaviour of the differential

pair. One way to solve this problem is to use an offset cancellation technique similar to the one discussed in the Chapter 3 of this thesis for CMOS analogue differential amplifier.

Another way of reducing the inherent offset voltage due to mismatches in the differential pair is to take the measurements two times, with the inputs of the differential pair interchanged between the measurements [43]. Taking the average of the two measurements ideally should eliminate the offset of the differential pair.

Figure 2-4 shows the schematic view of the technique proposed in [43]. The authors use several well-chosen measurement points (M1-M6) on the supply lines to find out what the currents through the various circuits inside the IC are.

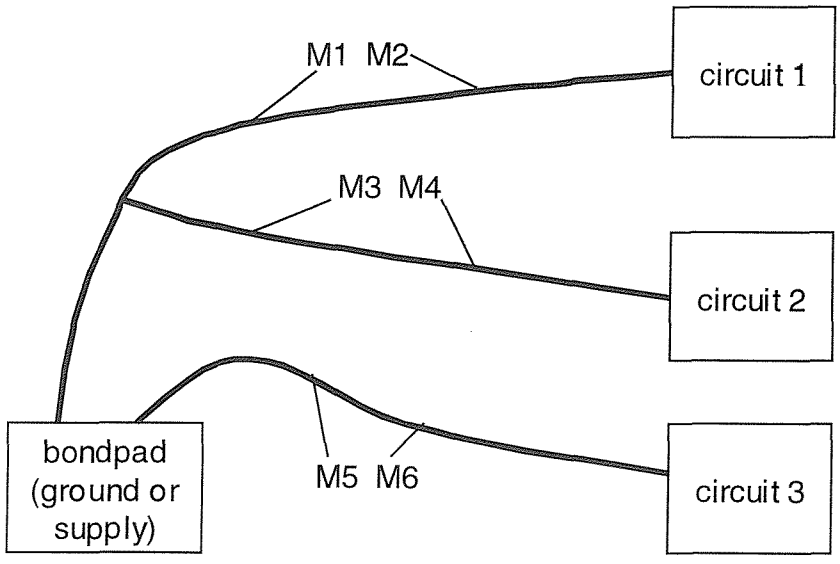


Figure 2-4. Basic configuration used in the technique presented in [43].

Figure 2-5 shows a way to combine all the measurement points to get the data to the outside world. For other possible implementations of the technique refer to [43].

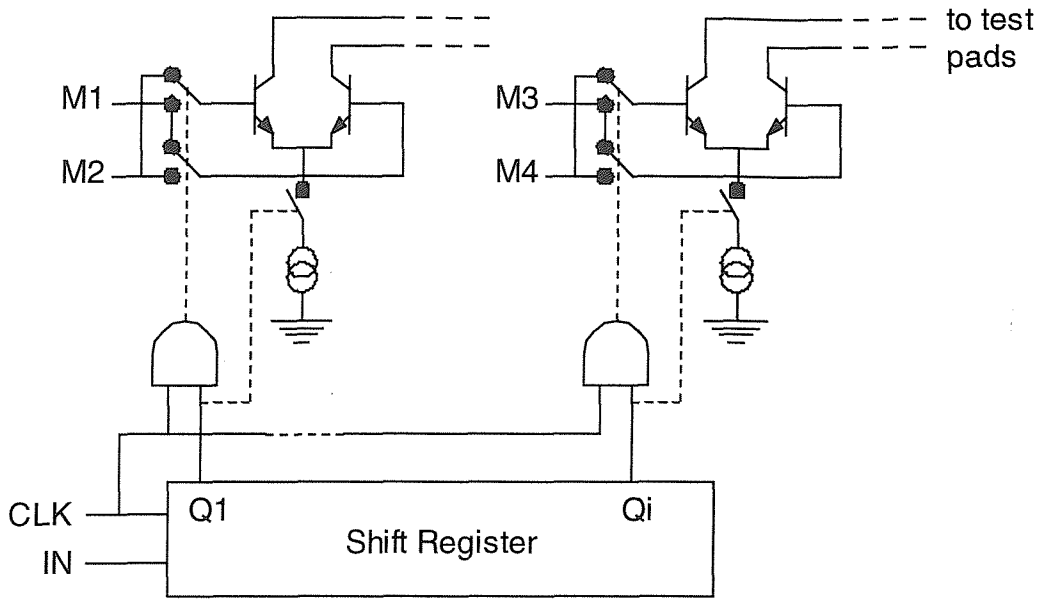


Figure 2-5. A possible implementation of current test technique suggested in [43].

The technique proposed in [43] was verified through the use of a one-chip TV processor IC. The IC was a mixed signal BiCMOS IC in which the TV signal was processed by analogue circuits. The control part of the IC was digital. The IC contained around 50,000 components of which 30 percent were in the analogue part. The whole IC draws 120 mA. The analogue part consumes 110 mA. The analogue part was divided into 150 basic functional cells of which the authors added 24 test measurement points. Each test measurement point monitored six basic functional cells on average with a current consumption of 4.6 mA.

The method proposed in [43] resulted in detection of 42.7% of the faulty ICs. The main advantage of the technique presented in [43] is that it is parallel to the supply lines, which means that the technique claims not to influence the normal operation of the IC.

2.2.4.2 Variable Supply Voltage Testing

The aim of the development of a test technique, which applies power supply levels outside the specified range, is to be able to detect the presence of defects which otherwise are not detectable at all or only by means of performance test. This technique is based on the fact that an IC still can perform certain functionality even outside the specified power supply range. Only the performance of the device might change for other supply levels [18]. Detailed investigation of this technique is presented in Chapter 4.

2.3 Design for Test (DfT)

An extract from an article published in 1998 on one of the world's leading microprocessor companies, Intel, web site is as follows. *"... A roadmap recently published by the Semiconductor Industry Association (SIA) predicts that by the year 2000, microprocessor transistor counts will exceed 21 million transistors. The same industry roadmap also predicts that by that time, microprocessors will utilize full flip-chip technology, instead of current wirebond assembly and packaging technology. Traditional fault isolation techniques using intensive e-beam probing and assembly code minimization as well as die frontside defect localization with liquid crystals or emission microscopy are no longer sufficient even for today's complex microprocessors like the Pentium and Pentium Pro microprocessors. Newer Failure Analysis (FA) techniques based on design-for-testability (DFT) and design-for-failure-analysis (DFFA) features have proven to be highly successful for the Pentium and Pentium Pro microprocessors, as evidenced by high analysis success rates (>90%) and short analysis throughput time. Without these new FA techniques*

that pinpoint the exact failing location, detection of sub-micron defects such as silicon dislocation, as shown in Figure 2-6, *is very difficult if not impossible...."* [44] According to another article (published on www.electronicnews.com web site on August 22, 2000), the newest (until the end of August 2000) Intel's microprocessor called Pentium 4 operating at 1.4GHz, contained 42 million transistors and ran with 400MHz of Rambus DRAM (RDRAM) [45].

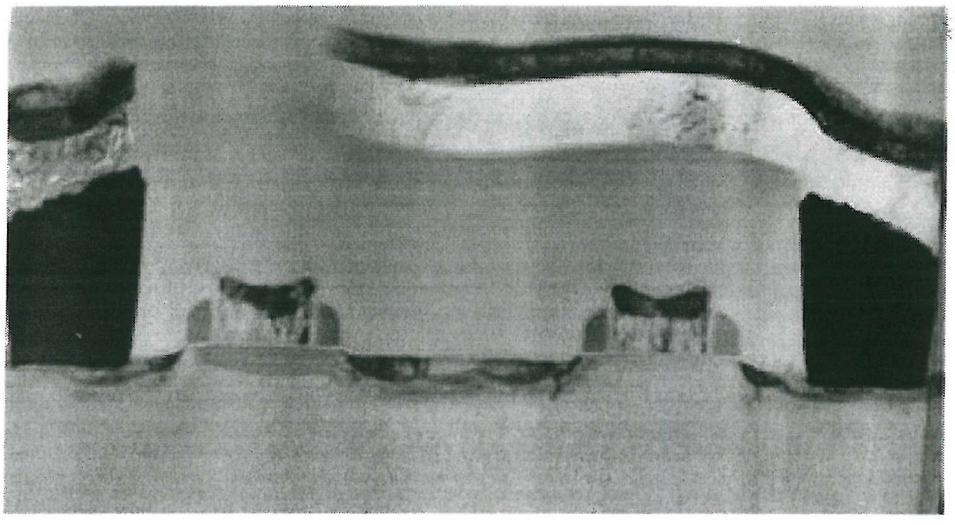


Figure 2-6. A TEM micrograph of silicon dislocation [44].

As technology shrinks smaller and smaller and the number of transistors increases, verification of functionality and bug identification has become a big problem. Microprobing the wafer is not very practical or possible in most cases in a technology smaller than 0.5 micron. Other forms of debugging and failure analysis such as E-Beam wafer probing are also limited. The ability to perform yield enhancements is also directly proportional to the debugging capability [46].

The complexity of circuit can be converted into costs associated with the testing process. There are several facets to this cost. Some of them are: the cost of the test pattern generation; the cost of the fault simulation and generation of the fault

location information; the cost of the test equipment; and the cost related to the testing process itself, namely the time required to detect and/or isolate a fault. Because these costs can be high and may even exceed design costs, it is important that they be kept within reasonable bounds. One way to accomplish this goal is the insertion of DfT into the design [47].

By increasing the testability of a circuit, it is implied that some functions of these costs are being reduced, though not necessarily each individual cost. For example, scan designs may lower the cost of test generation but increase the number of I/O pins, area, and test time [47].

Controllability, *observability*, and *predictability* are the three most important factors that determine the complexity of deriving a test for a circuit [1]. *Controllability* is the ability to establish a specific signal value at each node in a circuit by setting values on the circuit's inputs. *Observability* is the ability to determine the signal value at any node in a circuit by controlling the circuit's inputs and observing its outputs. The degree of a circuit's controllability and observability is often measured with respect to whether tests are generated randomly or deterministically using some ATPG algorithms. *Predictability* is the ability to obtain known output values in response to given input stimuli. Some factors affecting predictability are the initial state of a circuit, races, hazards, and free-running oscillators [1].

Technically, DfT is usually understood as a way of increasing controllability and observability of a circuit. There has been much discussion over the past many years of the tradeoffs involved in DfT decisions [47]. Things such as impact on product performance, design time, die size, wafer yield, test development time, fault coverage and product quality all enter into the equation when considering DfT

strategies. It is quite possible with today's complex SoC designs to have test development actually take longer than the functional circuit design effort itself. This is particularly true with the advent of reusable design elements; such as embedded memories and third party Intellectual Property (IP) cores [39].

2.3.1 Digital DfT

There are ad-hoc methods of implementing DfT, and these are still often used at the printed circuit board assembly level. Their use requires close and early design and test engineering communication, which is sometimes still difficult to achieve. But the ad-hoc methods are impractical for multi-million gate SoC designs [39], [45]-[47].

Most DfT techniques, such as scan, deal with either the re-synthesis of an existing design or the addition of extra hardware to the design. Most approaches, such as the DfT technique proposed in [49] for active analogue filters, require circuit modifications and affect such factors as area, I/O pins, and circuit delay. The values of these attributes usually increase when DfT techniques are employed. Hence, a critical balance exists between the amount of DfT to use and the gain achieved. Test engineers and design engineers usually disagree about the amount of DfT hardware to include in a design.

A design flow that does not consider DfT as one of the initial features of the design is going to have to become an iterative design flow that is time consuming and expensive. Today's most digital designs, such as microprocessors ([44], [48]), are done with HDLs, such as VHDL (VHSIC (Very High Speed Integrated Circuit) Hardware Description Language). Designers no longer have the schematic view that would let a test engineer spot testability problems. The HDLs lead directly to logic

synthesis in most cases. Therefore, testability problems are often not caught until post-synthesis. ATPG fails to produce high fault coverage tests. That means an iteration to redesign the chip so that it is much more testable [44], [47].

A much more TtM friendly approach involves simultaneous functional and testability circuitry design, including scan, BIST and boundary scan, and Register Transfer Level (RTL) testability rule checking and analysis prior to logic synthesis. Over ninety percent of testability rule and synthesis constraint violations can be caught at the RTL level using today's DfT Electronic Design Automation (EDA) technologies, with the remaining ten percent left for post-synthesis gate level double-checking [47].

Due to the existence of standard fault models, there are a number of DfT solutions possible for digital circuits. For example, there is a technique called *scan path*, used extensively in testing sequential logic circuits [35], [48]. During test mode the technique reconfigures storage elements, such as flip-flops, into a scan chain that is accessible via two pins, one at each end of the chain. This enables the status of flip-flops to be controlled and observed by clocking the chain. New values are shifted in as the contents of the flip-flops are shifted out. Then the circuit is put back into its normal operating mode and a single clock pulse is applied.

For large integrated circuit designs, more structured methods are required. Some of the techniques that have been in use are full scan, almost full scan, and partial scan [1], [45]-[47]. Full scan provides optimally testable circuits by replacing all of the latches and flip-flops in a design with scannable versions connected in series. This technique breaks all of the feedback loops in a design and provides vastly improved controllability and observability. The costs for full scan include some extra gate delays and some silicon overhead, but these costs are becoming

more and more negligible as we move into the deep submicron fabrication world. Where critical circuit paths really cannot stand the extra gate delays caused by full scan, partial or, more commonly, almost full scan can be used.

One major problem with inserting a DfT into a design is the limitation on the number of I/O pins available [35], [47]. Multiplexers are sometimes used in order to overcome this difficulty and to allow I/O pins to be reused in test mode. To control the multiplexers and other circuitry such as the scannable flip-flops, a test control structure could be included on-chip specifically to provide test control signals. This test control structure could consist of a shift register and some combinational logic. Test signals could then be set by scanning in values into the register. The advantage of this method is that it only requires a clock and a data-in pin to control many internal test signals [35].

Adding circuitry for DfT often impacts product performance by adding extra gate delays [35]. If done manually, DfT insertion can add considerable time to the design engineering task schedule. Every extra gate has the potential for increasing die size and therefore reducing wafer yield. Test generation for complex SoC designs has become increasingly problematical, as has the issue of grading the fault coverage of the generated tests [39]. Yet it is very unwise to ship devices to costumers without being very sure that the devices indeed function as advertised.

It is also critical that the DfT and built-in self-test (BIST) insertion, test pattern generation and fault simulation tasks be considered as an integral part of the design process right from the beginning. To do otherwise is going to result in an iteration of a design which will delay TtM considerably and which may also in many cases cost the producing company so many design-in losses that the product may never have the volume potential to make bringing it to market worthwhile at all [39].

A consortium of European and American companies formed the *Joint Test Access Group* (JTAG) in 1985. The group published a number of proposals between 1986 and 1988 which were formalised in 1990 as an IEEE standard 1149.1. This standard defines a test port and boundary scan architecture, which enables the observation and control of IC I/O pins. This solves the problem of interconnect testing, making the inputs and outputs of chips easily observable by inclusion of small digital circuits called *scan cells* in between all I/O pins and the core logic. The scan cells each contain two flip-flops and a number of multiplexers. In normal operation the boundary scan cells have almost no effect on the functionality of the circuit and allow signals to pass unaffected. In test mode it is possible to disconnect I/O pins from the core logic and digitally observe the inputs, or control the outputs of the scan cells. As the scan cells are connected to one another to form a scan chain, the observed values can then be shifted out and new control values shifted in at the same time. This DfT structure is very powerful as it allows testing of interconnects between chips. It also allows the control of all inputs and observation of all outputs to the core logic [35], [32].

Boundary scan requires four extra pins on an IC. Two are required for the scan chain being called test data-in (TDI) and test data-out (TDO). The remaining two are required for the state machine controller, where one is used as the test clock (TCK) and the other is used for selecting the test mode (TMS) [35], [32].

2.3.2 Analogue DfT

Following its successful application to digital circuits, DfT techniques for analogue circuits have gained tremendous importance in recent years [35], [49]-[51],

[55], [57]. Their goal is to reduce the cost of testing by introducing testability criteria early in the design stage.

The Mixed-Signal Test Bus, the mixed-signal extended version of the IEEE 1149.1 standard called IEEE 1149.4, has recently been approved as an official IEEE standard [52]. The IEEE 1149.4 has started to be used in semiconductor industry slowly. A reason for this limited use is that the standard is actually meant for board level test, in which case it is not suitable for most IC-level tests. Most of the crucial tests for analogue and mixed-signal modules are dynamic and high-speed, which fall outside the application domain of the IEEE 1149.4 [4].

Scan paths have been in use in digital testing for many years where they have been accepted as a standard method of enabling circuits to be tested. In digital circuits in order to test the combinational logic, scan-in scan-out (SISO) reconfigures the digital storage elements so that their contents may be observed and new values programmed into them. There have been a number of attempts to transfer this technique to the analogue domain [35].

Another problem is that analogue circuits have a time continuous behaviour in their operation whereas most digital circuits are clocked. This means that to implement an analogue DfT scan solution, the time continuous nature of a circuit's normal operation must be suspended. DC measurements have to be performed and all the circuitry to form the scan chain has to be added.

2.4 Built-In Self Test (BIST)

Built-In Self Test (BIST) is a design technique in which parts of a circuit are used to test the circuit itself [1].

SoC devices are becoming an enabling technology for a wide spectrum of embedded computing applications. The principal characteristics of these devices are that they contain some mixture of processor cores, embedded memory and a variety of mixed signal interfaces, and implement the majority of the functions that previously occupied an entire circuit board onto a single device [53]. Testing SoC devices is an important part of this new technology. Functions that were previously stand-alone can become deeply embedded with SoC and have to communicate with each other over sub-micron trace widths. Moreover, all the "outside world" interfaces are now on-chip, and can require signals as diverse as RF, video, audio, and digital for testing.

Therefore, BIST can play an important role in a component-level test of SoCs where access to the embedded virtual component is difficult or impossible. BIST is also valuable for devices that need to perform a diagnostic function upon them in the field. One such role is in mission-critical systems, where it is imperative for the unit to check itself on a regular basis and issue a warning if the system requires attention.

BIST techniques for digital circuits can generally be classified into two categories. On-line BIST includes concurrent and non-concurrent techniques. The second class, off-line BIST, includes functional and structural approaches.

Because of the success BIST has achieved as a robust, technology-independent solution for digital test, researchers have focused on BIST as a solution for mixed-signal test [54]-[57]. BIST is probably the most promising route to automating mixed-signal test generation. With BIST, the only test pattern to generate is one that comprises a few control signals to initiate BIST and a few control signals to read the result. This automated solution can dramatically reduce the time and effort to create a mixed-signal test and allows for test re-use.

Often, large mixed-signal ICs are "big-D, little-A;" that is, large amounts of digital circuitry are combined with a few analogue functions [58]. An example is a mixed-signal ASIC containing 100k digital logic gates or more, plus a Phase-Locked Loop (PLL), Analogue-to-Digital converter (ADC), Digital-to-Analogue converter (DAC), and possibly other analogue functions. The test program for the digital portion easily is generated using ATPG or BIST; devising tests for the analogue portion, however, is not at all easy or automatic. As a result, many companies now report that 80% of test effort is directed at the 20% of the chip area that is mixed-signal, significantly increasing their TtM, engineering costs, and risk of decreased quality during early production [58].

Interestingly, even totally digital ICs are starting to look more mixed-signal in nature, due to the increasing significance of delay faults and power-supply noise. Departing from purely digital behaviour contributes to the complexity and cost of test, which is growing as a percentage of product cost.

High-gate-count ICs tend to have high pin-counts, and high-pin-count testers are almost always digital. Statically testing each voltage level on a digital tester is practical and common for ADCs and DACs with 4-8 bits of resolution, but insufficient for high-speed converters. If the converter has higher resolution, then mixed-signal ATE becomes necessary for testing frequency-domain properties, raising the prospect of significant capital investment. PLLs can be tested using only digital signals, but highly accurate and continuously changing edge timing is required to evaluate frequency lock range, loop stability, and jitter.

The integration of third-party, mixed-signal, system-level macros was addressed by the Virtual Socket Interface Alliance (VSIA) Mixed-Signal Working Group. A very challenging and interesting test issue arises: How does an

IP provider deliver a mixed-signal macro with verifiable performance when the provider is not doing the testing? The macro may have very impressive specifications, but unless it can be tested on each IC, the specifications are worth little. Also, the quality of test access and the accuracy and capabilities of the IP purchaser's ATE affect the macro's specifications. These issues get worse at high frequencies. A panel discussion at the 1996 VLSI Test Symposium concluded that BIST eventually would be the only practical DfT method for high-frequency ICs, because of the difficulty of accessing signals without affecting the signals themselves [58].

Mixed-signal BIST has been proposed as a solution to these problems for many years, and has been the subject of many academic papers [54], [56], [59]-[62]. A few characteristics are common to many mixed-signal BIST proposals. Early approaches proposed reuse of digital random pattern generation and signature analysis for testing a DAC and ADC pair. Unfortunately, this approach is not noise tolerant, because a single, noise-induced bit error causes an incorrect signature [58].

Using analogue techniques introduces other difficulties. Connections to internal nodes of analogue filters and converters impact performance. For example, the added capacitance and resistance will decrease speed, add noise and increase cross-talk. The insertion of BIST requiring direct access to analogue circuitry is difficult to automate, both at the schematic and layout levels. Some analogue BIST require the added circuitry to have greater accuracy than the CUT; this is unrealistic because the CUT usually has been designed to the maximum accuracy available from the manufacturing process.

Industry needs mixed-signal BIST that offers the key features of digital BIST, including [58]:

- Excellent coverage of short and open circuit faults (95%);
- Essential ATE-independent at-speed testing for parametric fault coverage;
- Automated insertion into designs at the RTL (Register Transfer Level) (Verilog or VHDL) level;
- Insignificant impact on design style and performance;
- Insignificant impact on IC area and test time.

The nature of analogue testing imposes additional requirements beyond digital testing. These include:

- Accuracy that is relatively independent of normal processing variations (20%), so that test results are not affected by variations in the circuitry used for BIST;
- Precision that is relatively independent of noise, so that typical 50mV power-rail noise caused by thousands of logic gates switching has no affect on test repeatability;
- Measurement of key functional performances (non-linearity, gain bandwidth), so that datasheet specifications are verified directly to reduce the risk of defect escapes;
- Insignificant impact on test yield.

Note that adding patterns to a digital test almost always has no effect on yield, which is not true for analogue. Each analogue test added can cause a reduction in yield arising from noise, inaccuracy, or an imperfect correlation between the parameters tested and datasheet parameters.

2.4.1 Evaluating BIST

Many criteria need to be considered when evaluating digital, analogue, or mixed-signal BIST. Analogue and mixed-signal BIST require additional criteria.

Is mixed-signal BIST with these required features realistically feasible? Mixed-signal designers are well known for their ability to make every bit of performance from a given manufacturing process last as long as possible. Yet, a tester must be even more accurate than the circuit under test (CUT). So, the only way to get higher fault coverage in terms of testing the CUT is perhaps to use embedded digital techniques, rather than external analogue ones. Digital test techniques can achieve better fault coverage by using more signal processing time than the function being tested, and can do more processing without adding noise [58].

Digital methods are able to exploit the very few on-chip values that are completely insensitive to process variations, such as the frequency of the master clock and the supply voltage. Even these references are only accurate when averaged over time, because averaging is perhaps the only way to minimize the impact of noise. Fortunately, it is easy to accomplish averaging digitally. Digital circuit design also is able to exploit automated gate-level circuit synthesis from a HDL, and automated layout is available in many commercial software tools. Automated synthesis, or layout of general-purpose analogue circuitry, is still in development, however [58].

Parametric performance relative to functional specification limits is an essential output for industrial strength mixed-signal BIST. To address this, BIST, regardless of the test method used, must output results in terms of specifications that are meaningful to the designer, product engineer, and customer. If performance is reported any other way, then correlation to the specifications will need to be

determined, and correlation inaccuracies will result in yield loss or defect escapes. Therefore, a simple pass/fail output is not suitable for mixed-signal BIST. Product engineers need to monitor parametric performance to improve yield and prevent missed deliveries [58].

An analogue designer who has carefully optimised a design is unlikely to consider making any changes to accommodate testability. Re-simulation time can take weeks, and the performance impact may be intolerable or even unpredictable. Contrast this with digital BIST, where logic-optimisation tools can re-optimise designs to make the impact of adding BIST insignificant. Digital designers who must incorporate mixed-signal macros into their design must be able to address testability in a way consistent with their digital methods. Ideally, mixed-signal macros would appear only as a digital test problem to be addressed with a digital tool kit [58].

2.4.2 Progress in BIST

Digital methods certainly seem to offer the best, and possibly the only, route to industrial strength mixed-signal BIST. An all-digital BIST approach that exploits sigma-delta technology together with on-chip DSP has been described in [61]. The only analogue elements needed are a resistor and capacitor for the low-pass filtering of the sigma-delta bit stream, to obtain voltages between logic 0 and logic1. The stimulus is generated using an all-digital sigma-delta oscillator to produce a bit stream that, when filtered, gives a very pure sinusoid. This signal is the input for the ADC under test, and the ADC output is analysed by DSP routines such as Fourier Transforms to determine gain and harmonic distortion. The tested ADC can then be used with the same BIST hardware to address DACs and analogue circuitry. Of

course, if no programmable DSP is already on-chip, then adding a DSP just for BIST is expensive.

A technique that analyses a BIST scheme for mixed-signal SoC circuits in order to provide on-chip stimulus generation and response analysis has been reported in [62]. The technique uses the sigma-delta modulation principle in order to produce high-quality stimuli and obtain accurate measurements without the need of precise analogue circuitry. The authors used numerical simulations to validate their idea.

Another BIST approach was proposed in [59]. The technique is based on converting an analogue CUT into an oscillator. The authors connected a circuit's output to its input via a prescribed passive and/or active analogue circuit so that the loop's overall gain and phase caused oscillation. BIST is accomplished by detecting that oscillation occurs, and ensuring that its digitally measured frequency is correct. A drawback is that parametric faults that result in no oscillation prevent any diagnosis. The approach is attractive because of its simplicity, but does not measure any datasheet functions and relies on analogue fault simulation to verify fault coverage. As for any analogue BIST, imperfect correlation to datasheet functions will result in defect escapes or reduced test yield (which is a cost, just like area).

BIST method proposed in [60] feeds a pseudo-random bit stream directly into the filter under test, and observes the filter output with a comparator during prescribed digital time windows. The comparator's reference voltage must be reasonably accurate and noise-free. This method is only applicable to filters, and requires high clock frequencies to facilitate unrestricted choice of digital observation time windows.

In the next section, automatic test pattern generation algorithms for analogue circuits are summarised mostly from [63].

2.5 Automatic Test Pattern Generation for Analogue Circuits

Automatic test pattern generation (ATPG) algorithms for analogue circuits are classified into four classes [63]: functional test generation, structural test generation, test generation via automatic selection and ordering, and DfT based test generation. The third class is not ATPG but is included for completeness as a test reuse tool and also for its more immediate applicability in analogue test generation.

There are generally three types of functional ATPG algorithms for analogue circuits:

- In empirical functional ATPG, analogue test sets are generated empirically using the circuit specifications and the waveforms regularly in simulation such as DC, sine, step, square, ramp, etc. Time domain and frequency domain testing are employed depending on which parameters to measure.
- In functional analogue ATPG, the requirement to show test effectiveness using the empirical test sets as standard is the main driver. Many algorithms for functional analogue ATPG use mixed fault models such as catastrophic and parametric, where they assume that a linear circuit, when faulty, remains linear, which is appropriate for the parametric fault model but questionable for the catastrophic fault model.
- Several ATPG algorithms rely on sensitivity of an output or an observable signal with respect to either a component value or a collection of process parameters.

Structural ATPG algorithms can be classified into four main groups:

- Resistive based ATPG uses resistors to model structural faults.
- Linear programming algorithms as ATPG uses linearised fault macro models.
- I_{DD} based ATPG uses power supply current tests in generation of test sets for analogue circuits.
- Pseudorandom ATPG algorithm similar to the digital LFSR had been used to test analogue linear time-invariant circuits.

Given the existence of the functional empirical test set, it is reasonable to expect that a better test set may be derived or selected from this set once a fault model is established. The criteria for test set selection are quite simple: either to cover 100% of the selected faults, or to reduce the test cost by reducing the number of tests, to order tests to reduce test time, or to do all simultaneously.

Once a design modification is permitted to improve testability, the spectrum of algorithms for analogue ATPG becomes almost infinite. Since analogue DfT techniques have not been standardised, there are numerous DfT techniques, and for each there is an ATPG. The comparison in algorithm efficiency is thus extremely difficult since it needs to take into account the performance impact of the proposed DfT method: layout size, loading, test benefit, etc.

3 BUILT-IN CURRENT SENSORS FOR CURRENT BASED TEST

3.1 Introduction

Supply current monitoring has proved to be an effective method for testing digital and analogue circuits [6], [7]-[14]. It can be implemented either using Automatic Test Equipment (ATE) or Built-In Current Sensors (BICS). Using a BICS is more advantageous in terms of: test equipment costs; increasing the testing rate; improving the fault detectability and observability of the Circuit Under Test (CUT); higher current sensing resolution and avoiding the influence of Input/Output (I/O) currents which may dominate the chip's total current [7].

One way to increase the difference between the fault-free and faulty currents is to select a stimulus that causes the current flowing through the faulty components to dominate the supply current response. This technique has limitations, as it might not be possible to propagate the correct DC voltage or frequency of an AC stimulus to the CUT and there might be faults that remain undetected regardless of the stimulus.

A more direct approach is to partition the circuit into small blocks perhaps of similar complexity to an operational amplifier, and to measure the supply current

from each with a BICS. In this chapter different BICS approaches for analogue CMOS circuits are discussed and a new, process variation independent BICS design is proposed.

3.2 Built-In Current Sensors

Monitoring the supply current avoids the need to add intrusive circuitry that can load sensitive outputs or internal voltage nodes. Most of supply current monitoring techniques, however, suffer from poor resolution when measurements are taken off-chip [64]. The situation is worse for IDDQ testing of digital ICs because the large capacitance between the supply terminal and ground and associated test equipment must be discharged before a static DC measurement can be taken. One possible solution is to add one or more BICS [42].

There have been a number of BICS circuits proposed for digital applications [7], [9]-[12], [14] but most of them are not easily applicable to analogue circuits. Most of the sensors designed for digital applications cannot be used for monitoring analogue circuits since measurements for an analogue circuitry need to be taken continuously and analogue circuits have extremely non-linear transfer characteristics. The most common drawback perhaps with most of those sensors is that they require a large area for the realisation of the serial active element. This element has to sink the total current drawn by the CUT from the power supply. Researchers, therefore, have focused on designing BICS circuits for analogue circuits [65], [66].

Eckersall et al proposed using simple linear current mirrors monitoring each analogue macro within a two bit flash ADC [65]. In addition to the standard current

mirror, Renovell et al [66] added voltage monitoring at the output of the CUT. In [66], the authors proposed to build the Analogue Signature (*AS*), which is defined as the reduced set of data obtained by compressing the output response, by integrating the continuous time output response, ($O(t)$), between time t_1 and t_2 $\left(AS = \int_{t_1}^{t_n} O(t)dt \right)$. High fault coverage was quoted (98%) for the tested opamp.

Simple current mirror used in [65] and [66] will always have a significant DC voltage drop across the diode connected current mirror transistor that is serially connected between the CUT node, which connects the CUT to this serial transistor, and the supply rail. The main disadvantage of these sensors, therefore, is the performance degradation to the CUT due to this unwanted DC voltage drop.

One of the suitable sensors for analogue applications to minimise drawbacks with sensors used in [65] and [66] is the one that was proposed in [13] and implemented in [14], which is shown in Figure 3-1. This sensor is based on a series voltage regulator. A series voltage regulator with a very small voltage drop is modified by including an extra transistor M2, which monitors the gate source voltage of the main series transistor M1. A change in the supply current drawn by the CUT will be seen as a change in the gate voltage of M1 as it tries to maintain the set voltage drop. The gates of M1 and M2 are connected together.

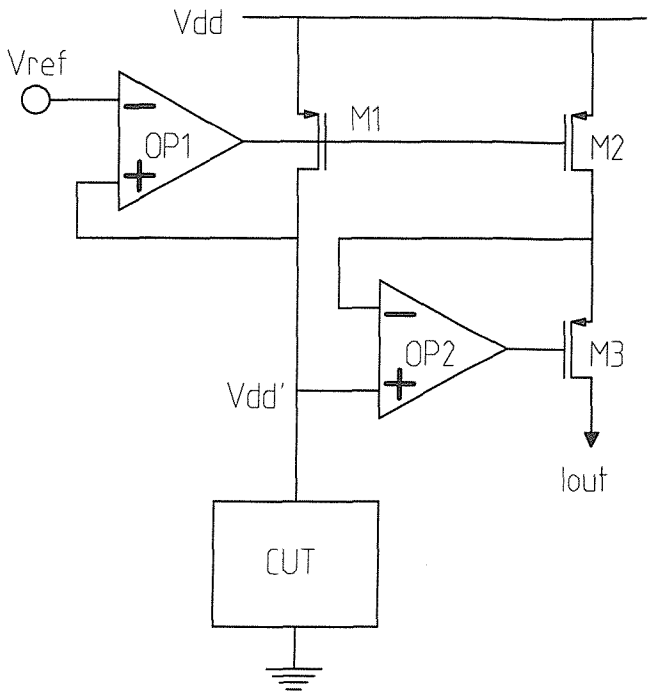


Figure 3-1. Series voltage regulator based BICS [14].

The current through this extra transistor M2 is therefore a copy of the supply current. OP2 and M3 are added to equalise the drain-source voltages of M1 and M2. One drawback with this sensor is the requirement for a very well defined reference voltage at the input of the comparator, OP1 in Figure 3-1.

Another drawback with the serial voltage regulator based BICS is the area required to realise the transistor M1. Since it is desirable to keep the voltage drop low (e.g. 25 mV for 2.5 V supply), the width/length ratio of M1 would have to be large enough to sink all the current (DC+AC) drawn from the supply by the CUT. As this current might be relatively large (10 mA for instance) for large and complex analogue circuits, the size of M1 would be excessively large. Therefore, a new sensor was proposed in [67] to overcome this disadvantage. This sensor is based on the shunt voltage regulator principle.

3.3 Shunt Voltage Regulator Based BICS

The shunt voltage regulator principle can be used to derive a voltage proportional to the dynamic current change in the CUT, which can then be used to monitor the CUT current. This means that, with shunt voltage regulator based BICS one can only monitor the dynamic current variation of the CUT rather than the absolute DC power supply current.

One can monitor the dynamic CUT current by using an active shunt element. The area of the shunt element will depend on the current variation that the CUT experiences during normal operation. For many analogue circuits, such as a two stage CMOS opamp, the power supply current variation can be less than one tenth of the quiescent bias current. Therefore, the size of the shunt transistor can be rather smaller than that of a series transistor of serial voltage regulator based BICS. Since the series element does not have to be an active device, it can easily be realised as a small value resistance, which will occupy a small silicon area.

How can one employ the shunt voltage regulator principle as a BICS design? The shunt voltage regulator based BICS was proposed in [67], and depicted in Figure 3-2. The transistor M1, which is driven by the opamp, forms the shunt element. The opamp compares V_{ref} to V_{div}, which is proportional to V_{drop}, the voltage across R_{drop}. The opamp and shunt transistor M1 would ideally stabilise the voltage across R_{drop} so a constant current flows through R_{drop}. Let us assume that when there is no change in the load current V_{div}=V_{ref}, hence the opamp output will be at the common-mode voltage level (assuming that the opamp is ideal). Now, if the load current changes by ΔI_{load}, V_{div} will change accordingly leading the output of the opamp to force the shunt transistor to draw an equal but opposite amount of current, ΔI_{shunt} (=-ΔI_{load}), from M1 based on the assumption that the opamp has a very high open-loop gain (80 dB for

instance). If transistors M1 and M2 share the same gate-source and drain-source voltages the current through M2 will be the same as the current through M1. The transistor M3 is included to act as a load to equalise the drain-source voltages for M1 and M2, which compensates for any difference in the current through M1 and M2. The current through M2 is mirrored with M4 and M5 and can be applied to an external pin for further processing. M4 and M5 can only have the same drain currents if they are identical and have the same drain-source current, which is not addressed in [67]. The latter issue with M4 and M5 can be solved either by using the cascode current mirror or the drain-source voltage equalisation technique used in [13] and [14].

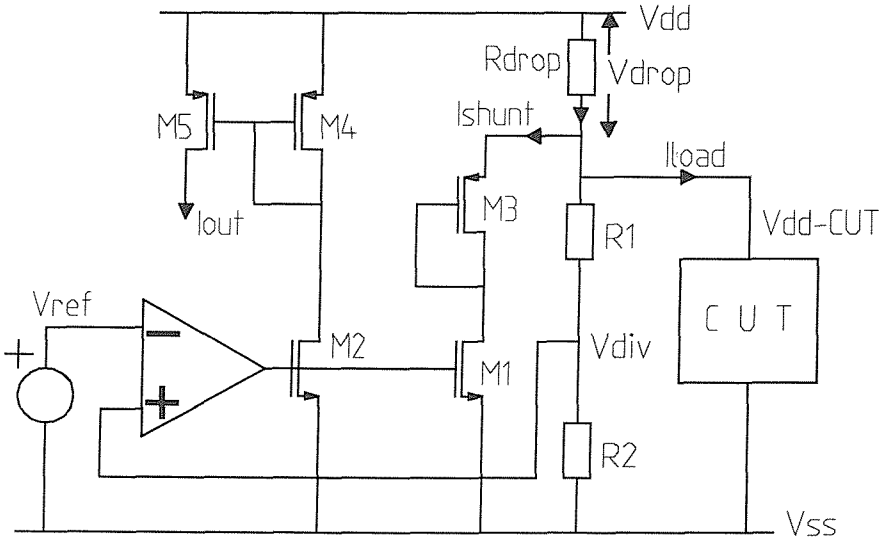


Figure 3-2. Shunt voltage regulator based BICS [67].

The monitored current can be converted directly to a voltage by connecting I_{out} to one terminal of a resistor with the other terminal connected to V_{ss}. If a number of sensors are needed and their outputs need to be individually selected, it is best if I_{out} is not converted to a voltage. Poor quality switches can then be used to multiplex

the outputs with no degradation in the signal fidelity. The output of the multiplexer can then be converted to a voltage and sent to an IC pin, if desired.

As proposed in [67] the value of R_{drop} is calculated from (3-1), and the width/length ratio of M1 is calculated from (3-2), respectively.

$$R_{drop} \cong \frac{V_{drop}}{I_{load} + |\Delta I_{load}|}, \qquad (3\text{-}1)$$

$$\Delta I_{shunt} = |\Delta I_{load}| = \frac{K}{2}\frac{W}{L}(V_{gs} - V_t)^2(1 + \lambda V_{ds}) \qquad (3\text{-}2)$$

where K is the transconductance of transistor M1 and λ is the channel length modulation constant. If it is assumed that V_{dd} is 5V and V_{ss} is 0V and V_{ref}=V_{div} then V_{gs} = V_{dd} / 2 (note that this is true only when the opamp is ideal). Let us assume that the example CUT draws 1mA DC supply current, 10μA AC supply current. If the maximum value of V_{drop} is chosen to be 50mV, then according to (3-1) the value of R_{drop} is going to be around 49.5Ω.

To calculate the width/length ratio of M1, let us assume that λ=0, the transconductance K is 8.9e-6 A/V^2, and the threshold voltage V_t is 1V. According to (3-2) the width/length ratio of the shunt transistor would be 1. In contrast, if a BICS based on the series regulator principle were used, the width-length ratio of the series transistor would be 101. Also note that the ratio of the shunt transistor is independent of V_{drop} and of the DC current drawn by the CUT. This is in contrast with the series regulator transistor; the width-length ratio of the series transistor is directly proportional to V_{drop} and the total current drawn from the supply.

Figure 3-3 shows the practical CMOS implementation of the shunt regulator based BICS. The voltage divider is realised by a chain of MOSFETs: MVPB; MVPA; MVNA and MVNB. The voltage reference circuit is also implemented through a transistor chain: MP1B; MP1A; MN1A and MN1B, which are connected between positive and negative supplies.

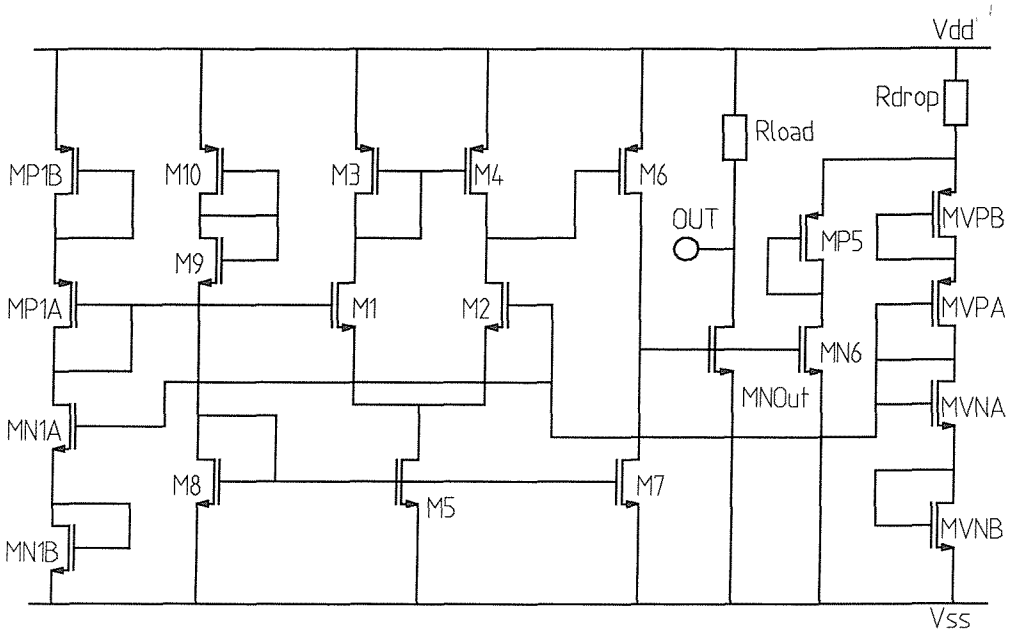


Figure 3-3. Simple 2-statge opamp based CMOS implementation of the shunt voltage regulator based BICS proposed in [67].

R_{drop} should have a small value in order to keep the voltage across it small. Therefore, it was realised in poly resistance in [69]. The main reason is that poly resist has a substantially smaller voltage dependence on the depletion-region width, compared to other types of resistances realised in CMOS technology.

Since it is desirable that R_{load} to be relatively large, it was realised as N-Well resistance in [69], which has a high resistance value per square. As can be seen from the Figure 3-3, the comparator was realised as a simple 2-stage operational amplifier [69], which is not desirable. The reason why a simple 2-stage opamp is not suitable for comparison of V_{ref} to V_{div} is discussed later in this chapter.

3.3.1 Simulation and Measurement Results

BICS circuit given in Figure 3-3 was analysed in [69] using HSPICE simulator [16]. Frequency response analysis had shown that the BICS had a 6 MHz frequency break point (-3dB frequency) [69]. The BICS circuit was laid out in [69] and post layout simulations were carried out. Post layout simulations had shown that the new BICS had 2.4 MHz –3dB frequency. The shift in the –3dB frequency is probably because of the parasitic capacitances and resistances coming from the interconnects. The circuit was fabricated in 2.4μ CMOS MIETEC technology along with an operational amplifier as the CUT [69].

The fabricated IC contains one BICS circuit and a 2-stage opamp circuit, which was designed to act as the CUT [69]. A number of measurements were carried out on the fabricated IC to confirm the simulation results. The voltage drop across the shunt resistor R_{drop} was found to be around 46 mV, which is very near to the expected V_{drop} value of 50 mV for 5V supply, for the number of packaged IC.

According to the simulation results, CUT opamp had 200μA static current with less than 20μA dynamic current consumption [69]. In order to confirm that BICS was monitoring the dynamic current consumption of CUT correctly, the AC voltage drop across R_{load} was measured and was divided by the value of R_{load}, which was designed to be 25KΩ. Unfortunately it was not possible to obtain the desired results by measurements. The realised BICS circuit did not work as accurately as expected. The reasons why the BICS did not function correctly are discussed below. Later, a new BICS design to overcome the drawbacks with the BICS proposed in [67] and realised in [69] is discussed.

The BICS circuit realised in [69] was designed to have a constant R_{drop} value to give a constant V_{drop} value of 50mV for one specific process parameter set during

the HSPICE simulations (nominal values of the process parameters were used in simulations done in [67] and [69]). Now, due to the process variations coming from imperfect fabrication conditions there will be deviations in the current drawn from the supply by the CUT for different process parameter sets. This will alter the voltage proportional to the dynamic CUT current, V_{div}, and the reference voltage, V_{ref}, to be different from the expected value found for the specific process parameter set used in HSPICE simulations. In the worst case, this will cause the output of the simple 2-stage opamp-based comparator to saturate to one of the supply voltages as the opamp is designed to have a high open-loop gain, hence the malfunctioning of the BICS circuit. Therefore, the realised BICS circuit [69] would perhaps function correctly if the IC manufacturing processes were perfect.

Another problem with the BICS mentioned above is that simple 2-stage opamp-based comparator will always have an inherent input-referred offset voltage due to the differential transistor pair used in the input stage of the opamp [68]. This offset voltage will add an unwanted voltage component to the inputs of the opamp, hence causing the output of the opamp to differ from the expected value. One way to solve this problem is to take the measurements at the output of the opamp two times while switching the inputs of opamp each time and add the two resulting voltages at the output, which should eliminate the inherent input-referred offset voltage. Another way of reducing offset voltage of a 2-stage opamp-based comparator is to use an offset reduction technique proposed in [70]. The latter technique is used in the new BICS design proposed in the next section due to its simplicity.

One more difficulty with the realised IC in [69] was that the controllability parameter was not considered during the design and layout of the circuit. This made

it impossible to observe some nodes within the fabricated IC during the measurements.

3.4 Process Variation Independent BICS

In this section, a new BICS design to overcome some of the drawbacks with the previously published BICS designs for analogue circuits is discussed. The main enhancement with this new BICS design is that it is process variation independent, which is very crucial for the correct operation of the BICS.

The new BICS design is similar in principle to the shunt voltage regulator BICS discussed in [67]. It is suitable for monitoring dynamic supply current variation of the CUT. One of two main improvements to the previous BICS circuit is; a new comparator design such that the inherent input referred offset voltage due to the differential pair used in the opamp circuit is minimised.

The second improvement is the removal of the reference voltage requirement at the input of the comparator. This is achieved simply by augmenting the previous design while keeping one input of the comparator at the common mode reference level. The dynamic current variation is sensed in the form of a voltage with a simple high pass filter network.

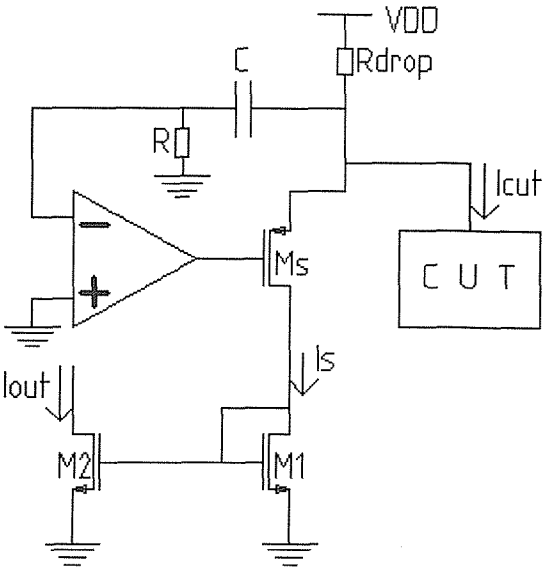


Figure 3-4. Process variation independent BICS design.

The simplified schematic view of the new design is given in Figure 3-4. As can be seen from the Figure 3-4, the simple high pass filter consists of a capacitance and a resistance.

The supply current variation of the CUT is considered only rather than the absolute DC current. The dynamic CUT supply current is monitored by using an active shunt element, Ms in Figure 3-4. The area of the shunt element depends on the supply current variation during normal operation.

Since the series element does not have to be an active device, it can be realised as a small value, small area resistance. The size of the series resistance is independent of the BICS and the CUT, unlike in [67], which means it can be kept as small as possible on the condition that the comparator has a high gain (80dB for instance). One could even use the parasitic resistance of the interconnect between the supply and the CUT.

The simplified operation of the BICS is as follows: The shunt element is formed as a PMOS transistor *Ms*, Figure 3-4. One input of the comparator is kept at

the common mode voltage, i.e. ground for designs with dual voltage supplies. If the current drawn from the supply by the CUT changes by ΔI_{cut} the voltage across R_{drop} will change proportionally. This voltage is then filtered by the RC high pass network and amplified by the comparator, which regulates the voltage to the CUT. This will cause *Ms* to draw the equal but opposite current from the supply. The size of *Ms* is determined by the gain of the comparator and by the dynamic CUT current. *Ms* is operating in the saturation region.

Since *M1* and *M2* form a simple current mirror, I_{out} will be the same as I_s if *M1* and *M2* are identical and have the same drain-source voltages (to suppress the effect of channel length modulation). Wide-swing cascode current mirror can be used in order to suppress channel length modulation effect [70]-[71], hence increase the accuracy of the mirrored current. The technique used in [13] and [14] can also be used to equalise the drain-source voltages of M1 and M2. I_{out} can be further processed for testing purposes.

In order to better understand the operation of the proposed BICS circuit, let us have a look at different sub-blocks of BICS design and the CUT circuit in more detail.

3.4.1 RC High Pass Filter

A simple RC high pass filter circuit is given in Figure 3-5. The transfer characteristic of the filter is depicted in Figure 3-6.

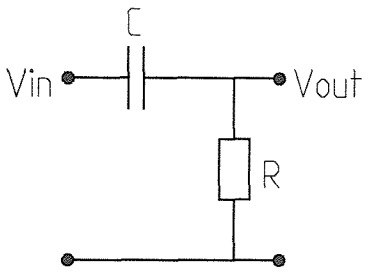


Figure 3-5. RC High pass filter circuit.

The transfer characteristic of the RC circuit of Figure 3-5 can be given as [72]

$$\left|\frac{V_{out}}{V_{in}}\right| = \frac{\omega RC}{\sqrt{1+\omega^2 R^2 C^2}} \qquad (3\text{-}3)$$

where $\omega = 2\pi f$ and f is the frequency.

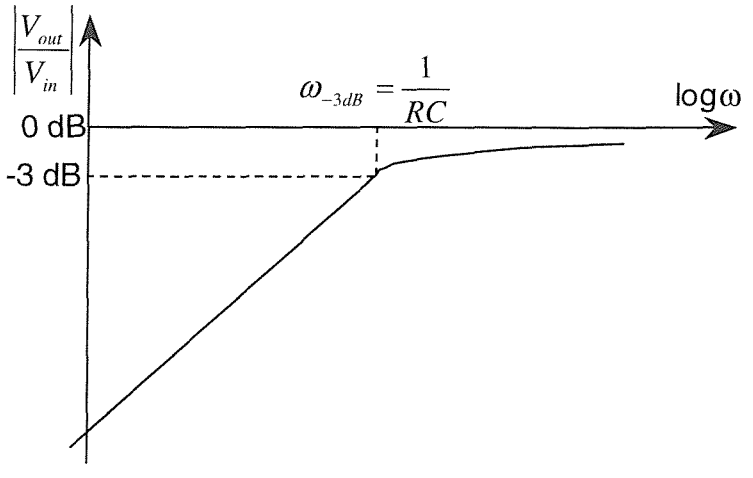


Figure 3-6. Transfer curve for the RC circuit given in Figure 3-5.

The circuit given in Figure 3-5 is an AC voltage divider with an output that falls off at low frequencies at the rate of 6 dB per octave [72]. Therefore, the circuit

will eliminate the DC component of its input voltage. This is desirable, the dynamic change in the supply voltage of the CUT needs to be captured.

As one of the goals is to keep the area used for BICS circuit minimal, the value of the capacitance was chosen to be 1pF, which is reasonable in terms of silicon area for the technology used in this thesis (0.8μm AMS CYE CMOS (2.5-5.5V, p-sub, 2-metal, 2-poly) [15]). The value of the resistance can then be chosen to be 100 MΩ (to give $\omega_{-3dB} = 10^4 rad/\sec (or\ f \cong 1.6\,kHz)$), which is enough to filter the DC component of the filter input signal). The high resistance coming from the input of the differential transistor pair used in the comparator circuit can be utilized or R can be implemented using switched capacitor technique proposed in [70].

3.4.2 Switched-Capacitor Comparator

The simple 2-stage opamp circuit used as the comparator in the previous BICS circuits [14], [67] is not efficient due to the inherent offset voltage coming from the mismatches in the transistors within the differential pair used in the opamp circuit. Therefore, a new comparator with reduced input offset voltage is required for the accurate operation of the BICS.

A new switched-capacitor, input offset reduced comparator circuit is therefore proposed. The comparator is based on a 2-stage Miller-compensated operational amplifier. The offset reduction technique proposed in [70] is used with a slight modification.

The comparator circuit is given in Figure 3-7 and CMOS realisation is given in Figure 3-8. The clock frequency for the switches in the comparator should be kept at least 10 times smaller than the normal CUT operation frequency in order not to affect the accurate operation expected from the comparator, that is the comparator

have enough time to do comparison and allow us to monitor the dynamic CUT current. Another important thing to bear in mind is that because the junction and subthreshold leakage of the switches eventually corrupts the correction voltage stored across C1, periodic refreshing of this voltage is required [70]. This might affect the correct operation of the comparator for applications requiring very low frequencies.

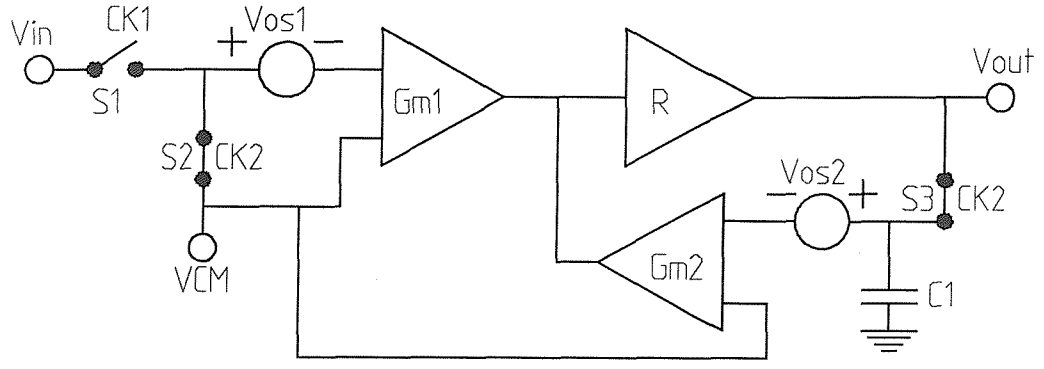


Figure 3-7. Switched-capacitor input offset reduced comparator.

V_{CM} in Figure 3-7 represents the common-mode voltage level for the circuit (V_{CM} = 0 V for designs with symmetrical dual supplies). G_{m1} stage (M1, M2 and M5 in Figure 3-8) and G_{m2} stage (M10, M11 and M12 in Figure 3-8) are ideal differential pairs with no input offset voltage, R stage (M6 and M7 in Figure 3-8) represents a transconductance amplifier. V_{os1} and V_{os2} represent the inherent offset voltages due to G_{m1} and G_{m2} respectively. V_{in} represents the input signal. S1, S2, and S3 represent switches, which are clocked with CK1, CK2, and CK2 respectively. CK1 and CK2 are non-overlapping clocks. C1 is used as a storage element.

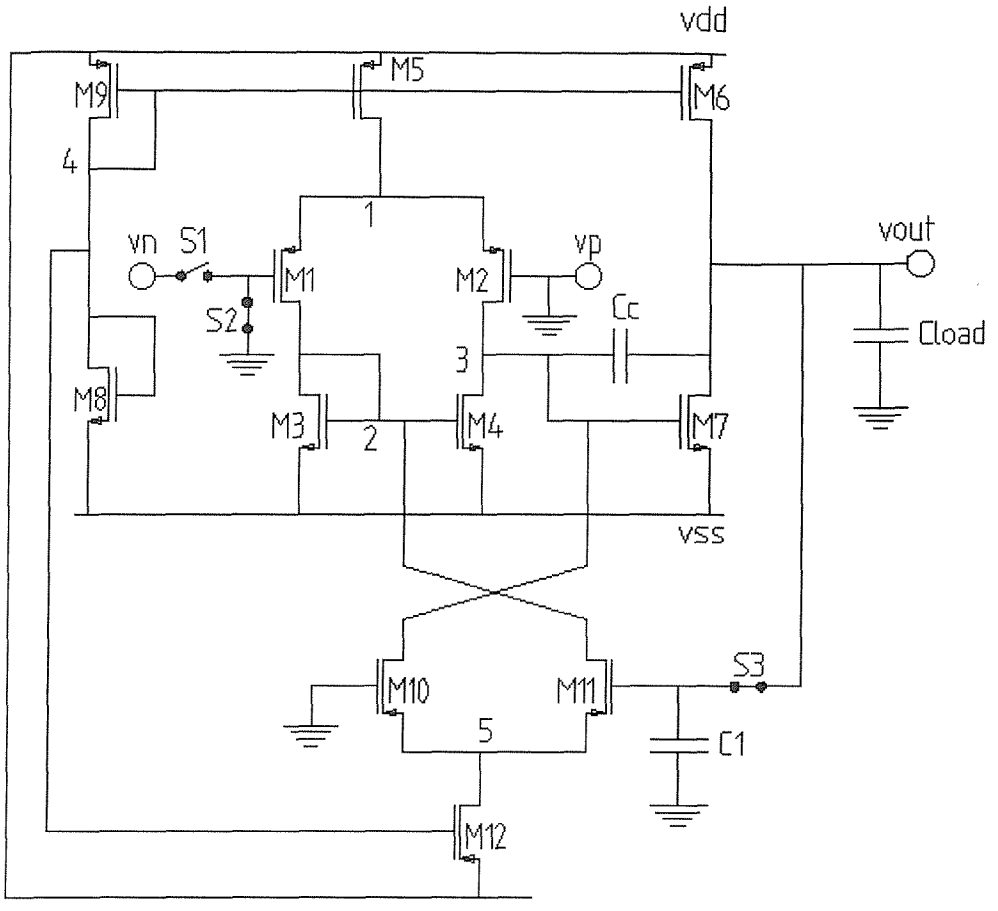


Figure 3-8. CMOS realisation of the comparator given in Figure 3-7.

G_{m2} is used as an auxiliary amplifier in a negative feedback configuration during the offset cancellation mode. G_{m2} simply amplifies the voltage stored across C1 and subtracts the result from the output of G_{m1}.

Let us assume S1 is off and S2 and S3 are on. Now, from Figure 3-7 one can write the following

$$V_{out} = [-G_{m1}V_{os1} - G_{m2}(V_{out} - V_{os2})]R, \qquad (3\text{-}4)$$

and the output voltage can now be written as

$$V_{out} = \frac{-G_{m1}RV_{os1} + G_{m2}RV_{os2}}{1+G_{m2}R}. \qquad (3\text{-}5)$$

This voltage is stored on C1 after S3 turns off. The overall offset voltage referred to the main input for the comparator is therefore given by

$$V_{OS,total} = \frac{V_{out}}{G_{m1}R} = -\frac{V_{os1}}{1+G_{m2}R} + \frac{G_{m2}}{G_{m1}}\frac{V_{os2}}{1+G_{m2}R}. \qquad (3\text{-}6)$$

As can be seen from (3-6) the overall offset voltage is reduced by the feedback loop gain of $(1 + G_{m2}R)$. Note that there is an additional component to the comparator's overall offset voltage due to G_{m2}. As can be seen from (3-6) this component is also reduced by the feedback loop gain. If it is assumed that G_{m2} is smaller than or equal to G_{m1} then this additional component does not have a very big effect on the normal operation of the comparator.

3.4.2.1 Non-overlapping clock generation

For the correct operation of the comparator circuit given in Figure 3-8 suitable clock waveforms for the switches are needed.

Switch S1 is clocked with CK1 and switches S2-S3 are clocked with CK2, where CK1 and CK2 are non-overlapping clocks in order to reduce the charge injection errors [70]-[71]. CK1 and CK2 clock waveforms are given in the time domain as depicted in Figure 3-9, where T1 represents the period of the CK1 and T2 represents the period of the CK2.

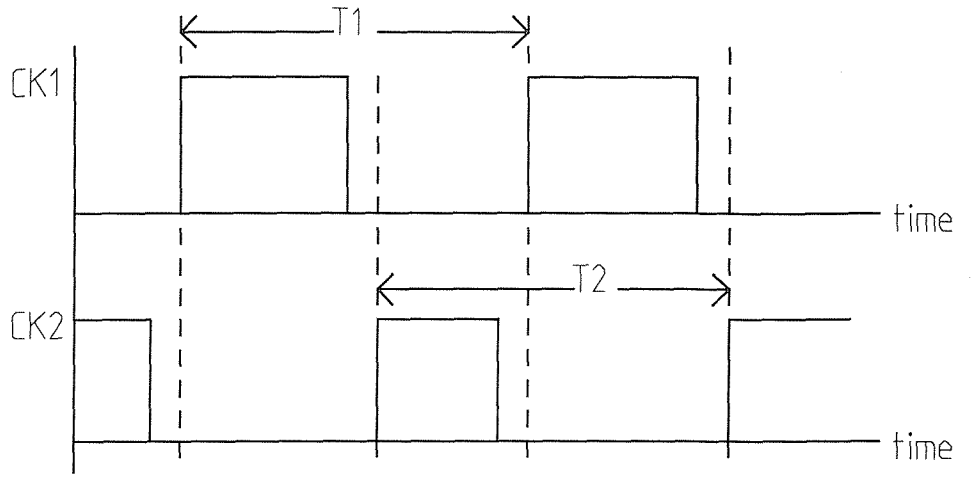


Figure 3-9. Non-overlapping clocks.

The technique proposed in [71] and depicted in Figure 3-10 can be used in order to generate the non-overlapping clocks. By using the circuit given in Figure 3-10, it is ensured [71] that there will be two inverter delays between CK1 and CK2 clock waveforms, where CKin represents the input clock signal. If this delay is not adequate one can use an even number of inverters to the inputs of the NOR gates shown in Figure 3-10 in order to reach the desired level of delay for the non-overlapped clock waveforms.

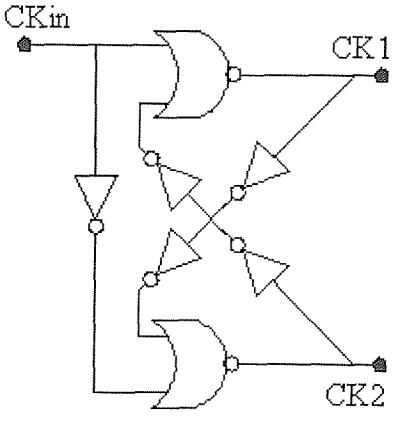


Figure 3-10. Non-overlapping clocks generator [71].

3.4.2.2 CMOS Switch

In order to further reduce channel charge injection [70], the CMOS switch depicted in Figure 3-11 is used in the comparator circuit.

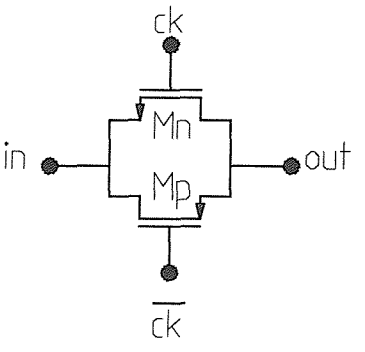


Figure 3-11. CMOS Switch.

For the correct operation of the CMOS switch complementary clock waveforms are required. It is important that NMOS and PMOS transistors (Mn and Mp in Figure 3-11) turn on and off simultaneously in order to ensure that the CMOS switch does not have any distortion on its output. If it is assumed that the NMOS device in Figure 3-11 turns off Δt time earlier than the PMOS device, then the output voltage tends to track the input for the remaining Δt time with a large, input-dependent time constant leading to a distortion, as shown in Figure 3-12 [70].

Therefore it is important to generate complementary clocks as depicted in Figure 3-13 for the accurate operation of the CMOS switch with no distortion at its output.

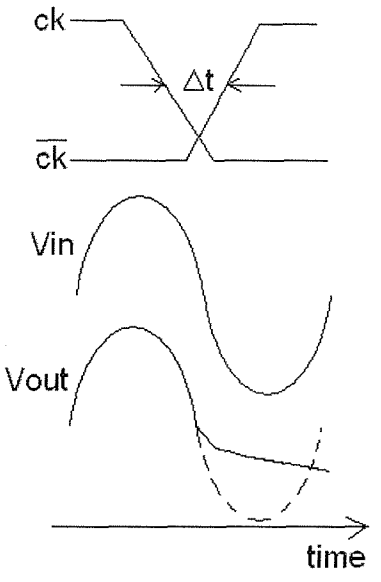


Figure 3-12. Possible distortion on the output of the CMOS switch due to a delay between clocks [70].

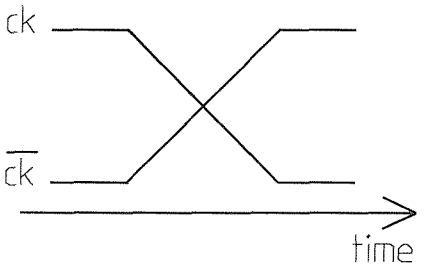


Figure 3-13. Complementary clock waveforms.

One way to produce equally delayed complementary clocks is to use the simple technique proposed in [70] and depicted in Figure 3-14. The CMOS implementation of Figure 3-14 is given in Figure 3-15. The MOS transistors used in the transmission gate in Figure 3-14 can be sized such that the transmission gate has the same delay as the inverter (assuming all the inverters have the same delay).

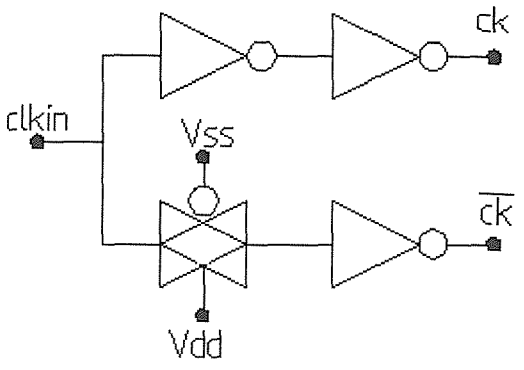


Figure 3-14. Equally-delayed complementary clocks generator [70].

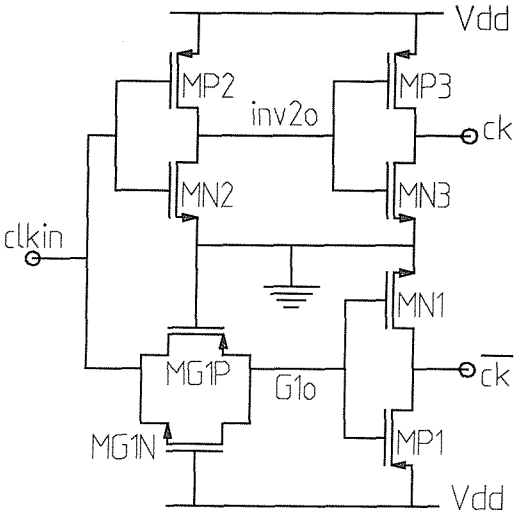


Figure 3-15. CMOS implementation of the circuit given in Figure 3-14.

3.4.2.3 Simulation Results for the Comparator

Both the comparator and the equally delayed non-overlapping clock generator circuits were designed in 0.8μm AMS CYE CMOS (2.5-5.5V, p-sub, 2-metal, 2-poly) [15] technology. The simulations were done using HSPICE with BSIM 3v3 [17] model parameters. In order to see the behaviour of the comparator under the process variations the simulations were carried out for three different

process parameter sets: worst case power (WP), typical mean (TM) and worst case speed (WS).

Before giving the simulation results for different process parameter sets one needs to have a look at the generated clock waveforms needed for the comparator circuit given in Figure 3-8.

Using circuits given in Figure 3-10 and Figure 3-14 suitable clock waveforms needed for the comparator circuit as depicted in Figure 3-16 are generated. In Figure 3-17 expanded clock waveforms are presented. It can be seen that there is a 2ns delay between ck1 and ck2, which makes them non-overlapping.

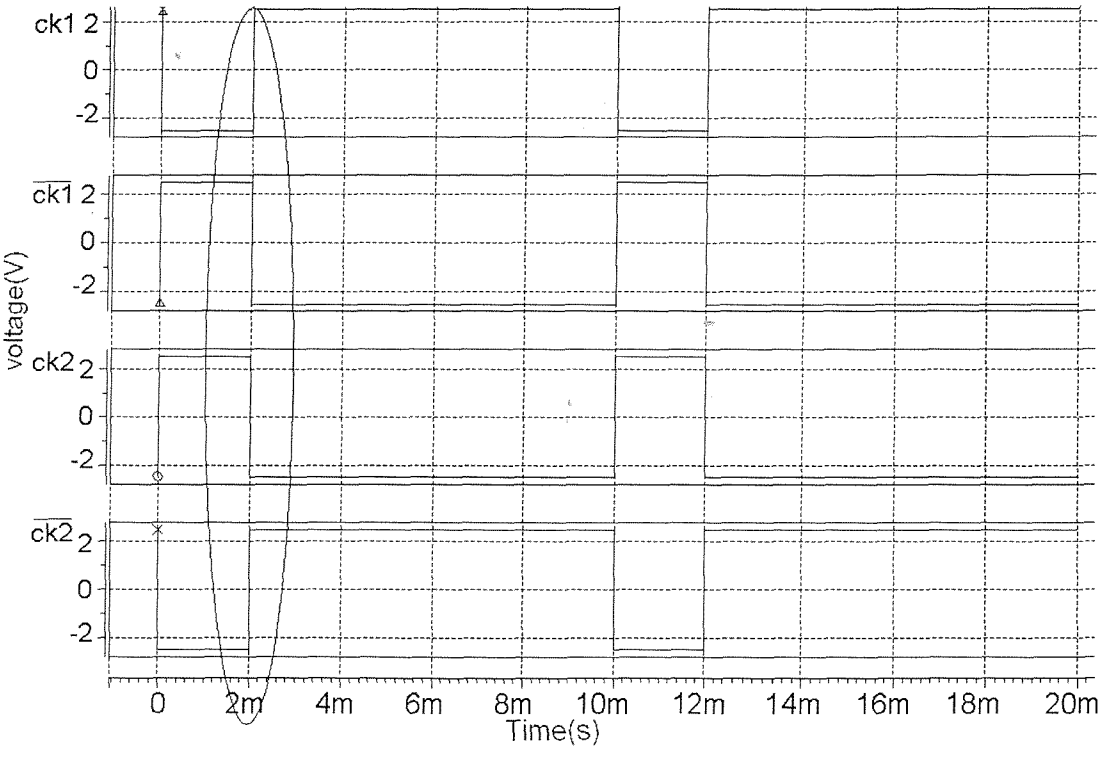


Figure 3-16. Suitable clock waveforms for the comparator circuit.

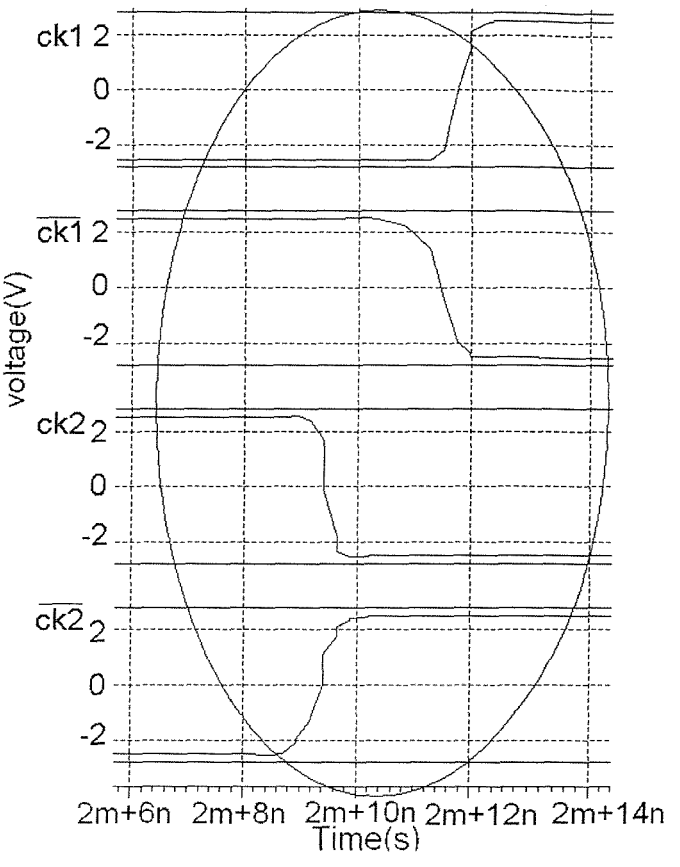


Figure 3-17. Expanded view of Figure 3-16.

In Figure 3-18, Figure 3-19 and Figure 3-20 comparator input versus comparator output voltage are given for TM, WS and WP process parameter sets respectively. The comparator input was driven by a sine wave with 300μV as shown in Figure 3-18, Figure 3-19 and Figure 3-20. It can be seen from the figures that the output of the comparator correctly tracks the input without saturating to one of the supply voltages. Therefore, one can use this comparator in applications such as the proposed BICS circuit given in Figure 3-4.

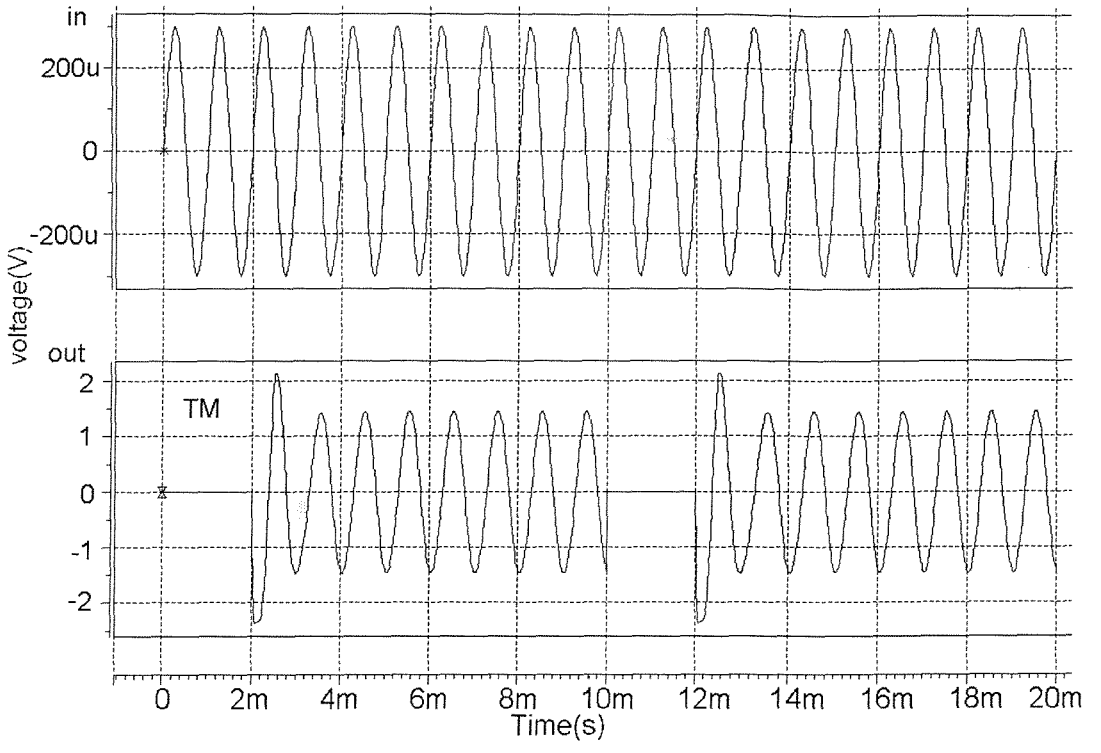


Figure 3-18. Comparator input and output voltages for TM parameter set.

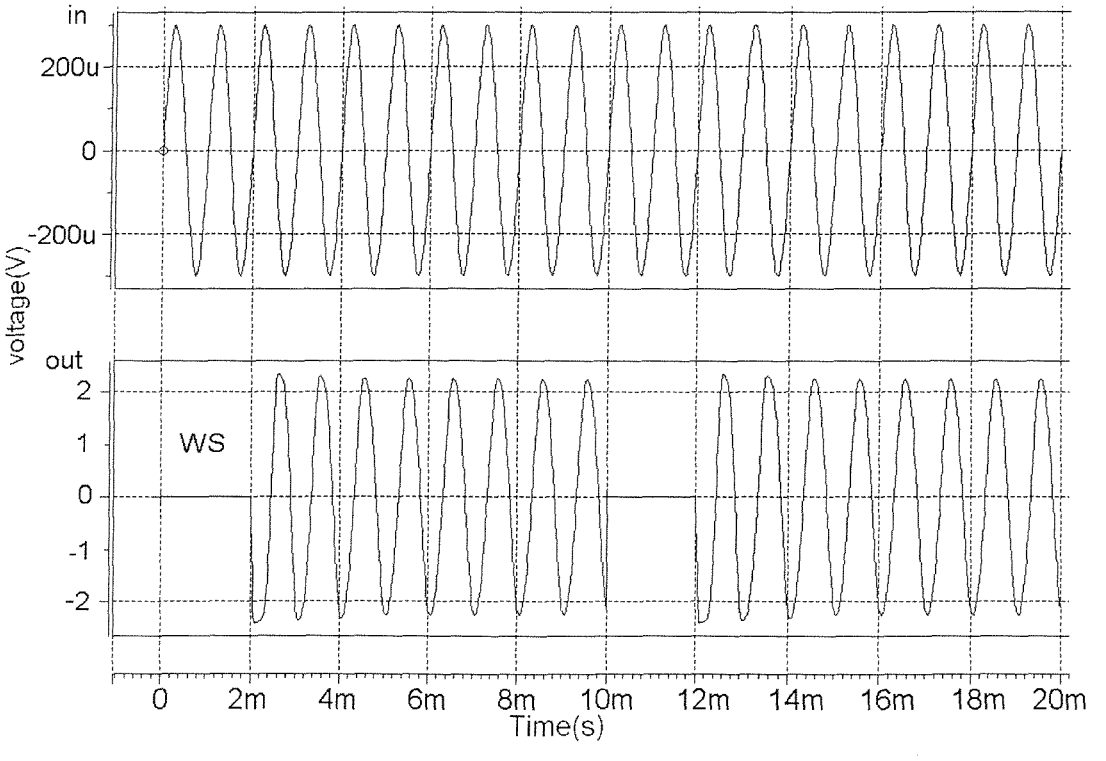


Figure 3-19. Comparator input and output voltages for WS parameter set.

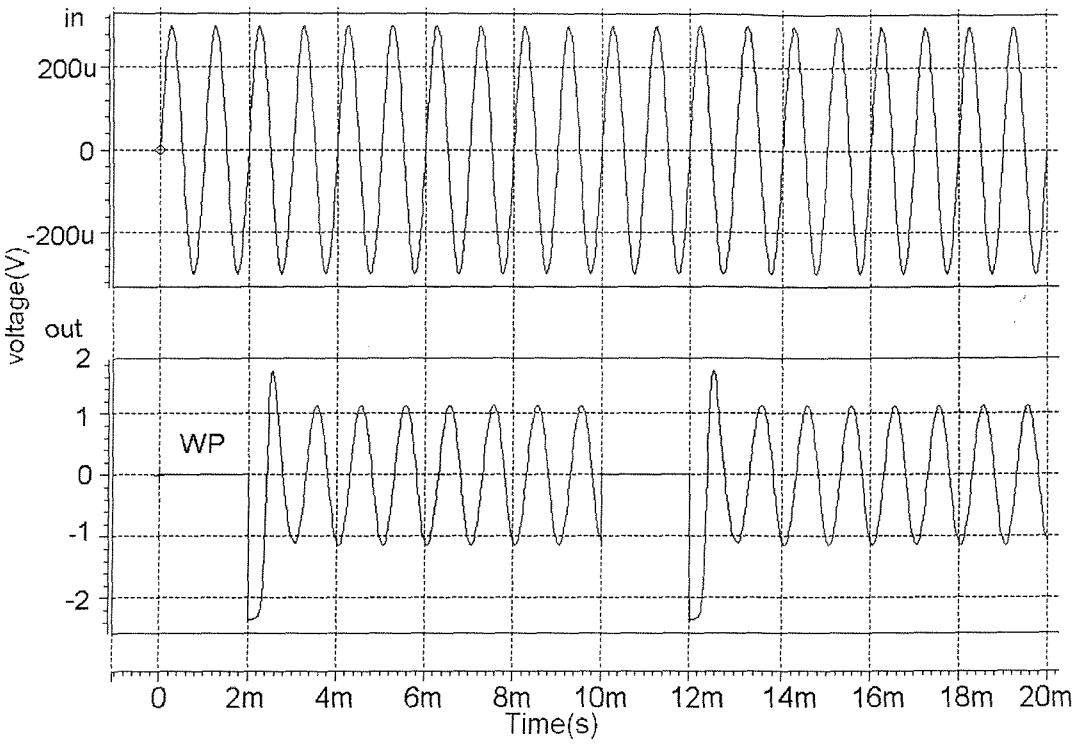


Figure 3-20. Comparator input and output voltages for WP parameter set.

For the TM parameter set the offset voltage seen at the output of the comparator, while it is in offset reduction mode, is only about –1mV. For WS and WP parameter sets the values for the offset voltages are -0.645mV and -1.69mV respectively. It is shown that the worst output referred offset voltage is obtained with the WP parameter set, which is less than 2mV in amplitude. This is very important for the accurate operation of the comparator circuit especially when high gain is needed.

3.4.3 State Variable Filter as the CUT

The state-variable filter from the IEEE Mixed-Signal Benchmark Suite [73] is used for the CUT. The circuit schematic is depicted in Figure 3-21. The filter circuit was redesigned in 0.8μm AMS CYE CMOS (2.5-5.5V, p-sub, 2-metal, 2-poly) technology. Simulations were carried out in HSPICE with BSIM 3v3 model

parameters during the design characterisation. Both the CUT and the BICS operate at ±2.5V supply voltage.

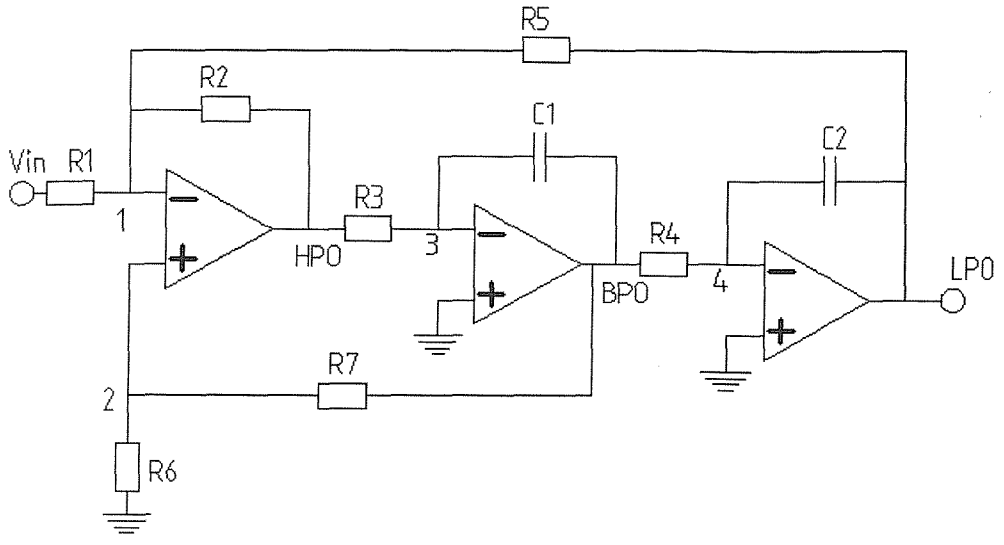


Figure 3-21. Continuous-time state-variable filter [73].

The transfer function of the second-order state-variable filter for its band-pass output (BPO in Figure 3-21) can be given by [73]

$$V_{BPO} = K \frac{\frac{s^2}{R3C1}}{s^2 + \frac{1}{QR3C1}s + \frac{1}{R3C1R4C2}} \qquad (3\text{-}7)$$

where

$$K = \frac{R2}{R1} \cong \frac{R5}{R1} \text{ if } R2 = R5 \qquad (3\text{-}8)$$

and

$$Q = \frac{(R6 + R7)}{(1 + 2K)R6} \qquad (3\text{-}9)$$

where K stands for the filter gain and Q is the filter quality factor. The central frequency of the BPO can be given by [73]

$$f_c = \frac{1}{2\pi\sqrt{R3C1R4C2}}. \qquad (3\text{-}10)$$

where $R1=R2=R3=R4$=R5=10kΩ, R6=3 kΩ, $R7$=7kΩ and $C1=C2$=20nF K=1, Q=1.11 and f_c=795Hz for the BPO. For simplicity, the resistance and capacitance values are chosen to be the same as the values given in [73]. Having examined the BICS circuit and the CUT in detail let us now have a look at the simulation results obtained for the BICS circuit.

3.4.4 Simulation Results for the BICS

The input stimulus to the filter circuit was chosen to be a 1.3V sinusoidal wave signal at 1KHz. As mentioned before, R and C were chosen to be 100MΩ and 1pF respectively. The serial resistance (*Rdrop* in Figure 3-4) value was chosen to be 20Ω. The size of *Ms* is 2μm/1μm. To show the process independence of the BICS, simulations for three different process parameter sets were carried out: worst case power (WP), typical mean (TM) and worst case speed (WS). The greatest supply voltage degradation to the CUT was 35mV for WP parameter set yielding to 2.465V positive supply voltage level. This does not affect the normal operation of the CUT significantly.

The simulation results for different process parameter sets are shown in Figure 3-22, Figure 3-23 and Figure 3-24.

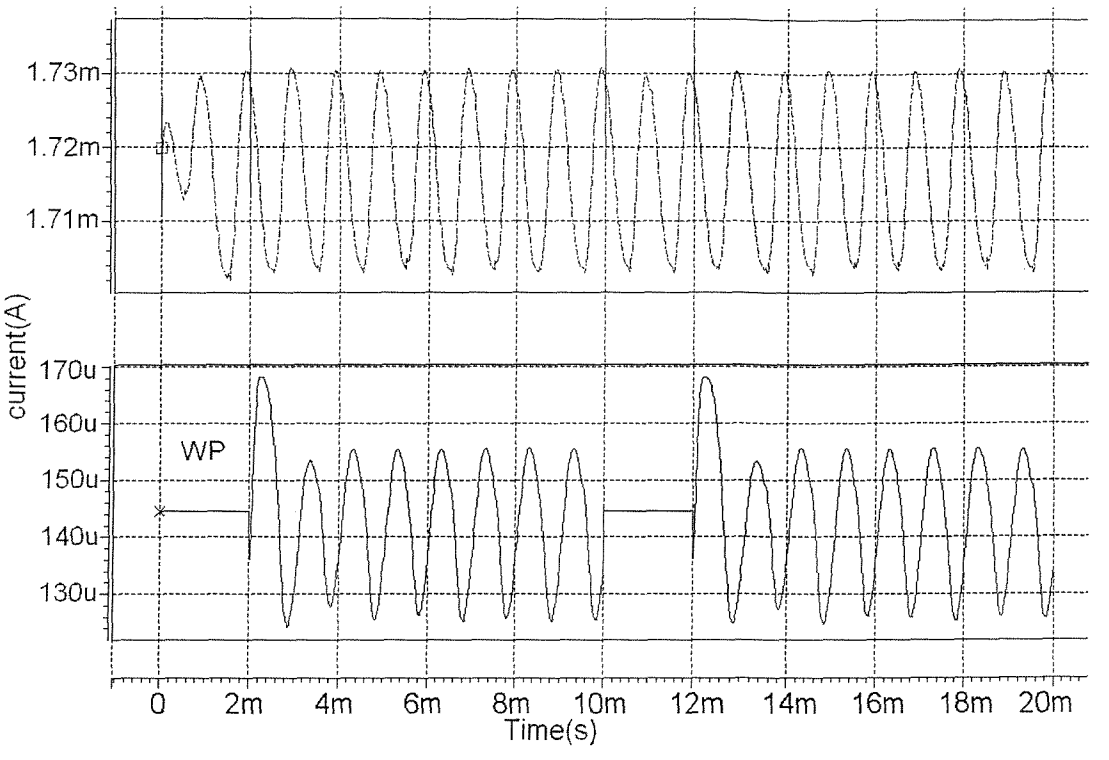


Figure 3-22. The current drawn by the CUT (top) and the monitored current with the BICS for WP parameter set.

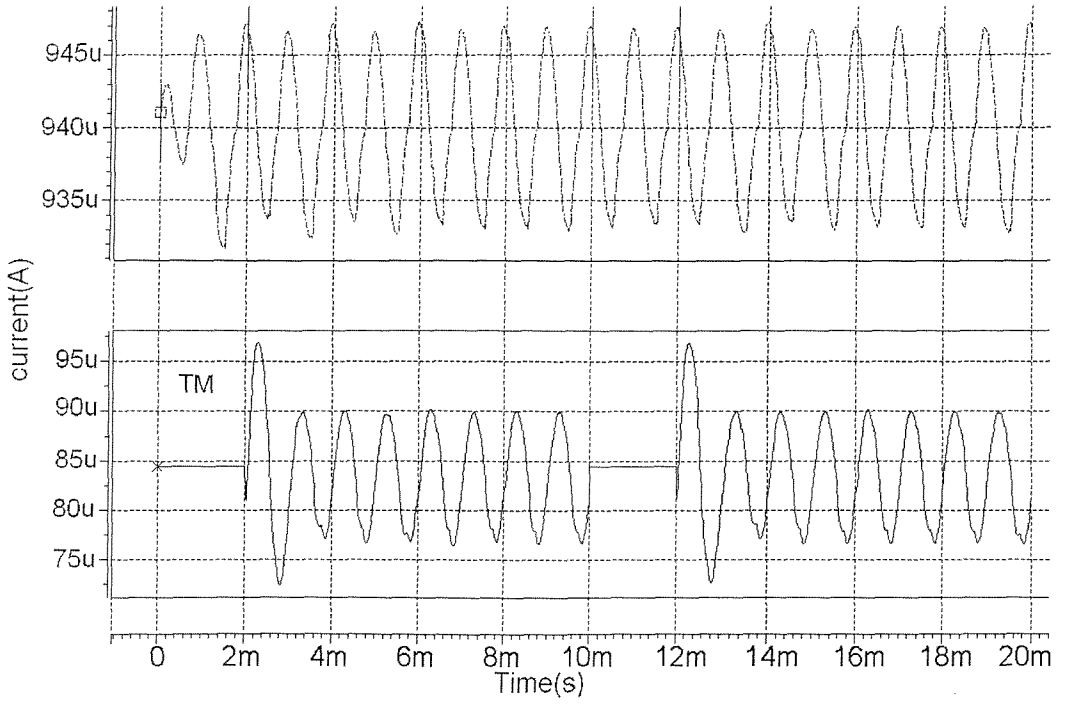


Figure 3-23. The current drawn by the CUT (top) and the monitored current with the BICS for TM parameter set.

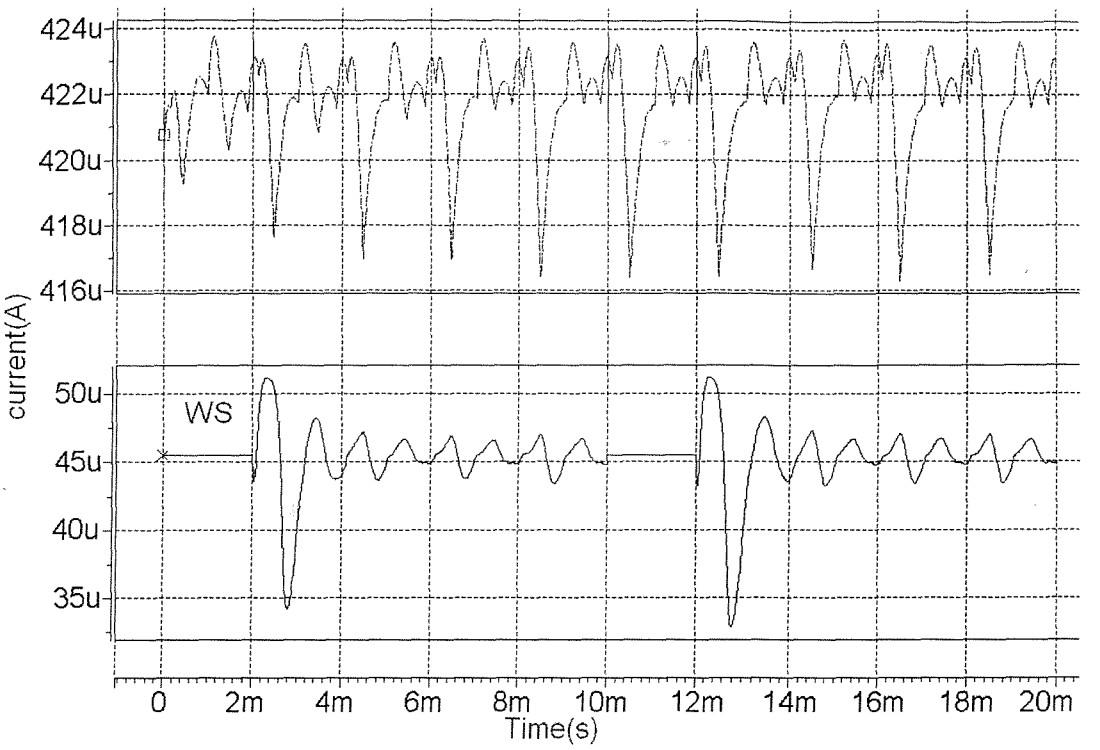


Figure 3-24. The current drawn by the CUT (top) and the monitored current with the BICS for WS parameter set.

As can be seen from Figure 3-22, Figure 3-23 and Figure 3-24, the current sensed with the BICS copies the dynamic CUT current variation accurately. The dynamic peak-to-peak current amplitude for the TM parameter set is 13.3μA, for the WP parameter set 30μA and for the WS parameter set 3μA.

In Figure 3-25, the monitored current versus comparator output voltage is given for the TM parameter set. It can be seen from the figure that the output of the comparator has a $0.8V_{peak\text{-}to\text{-}peak}$ change with respect to 13.3μA dynamic peak-to-peak current change due to the CUT.

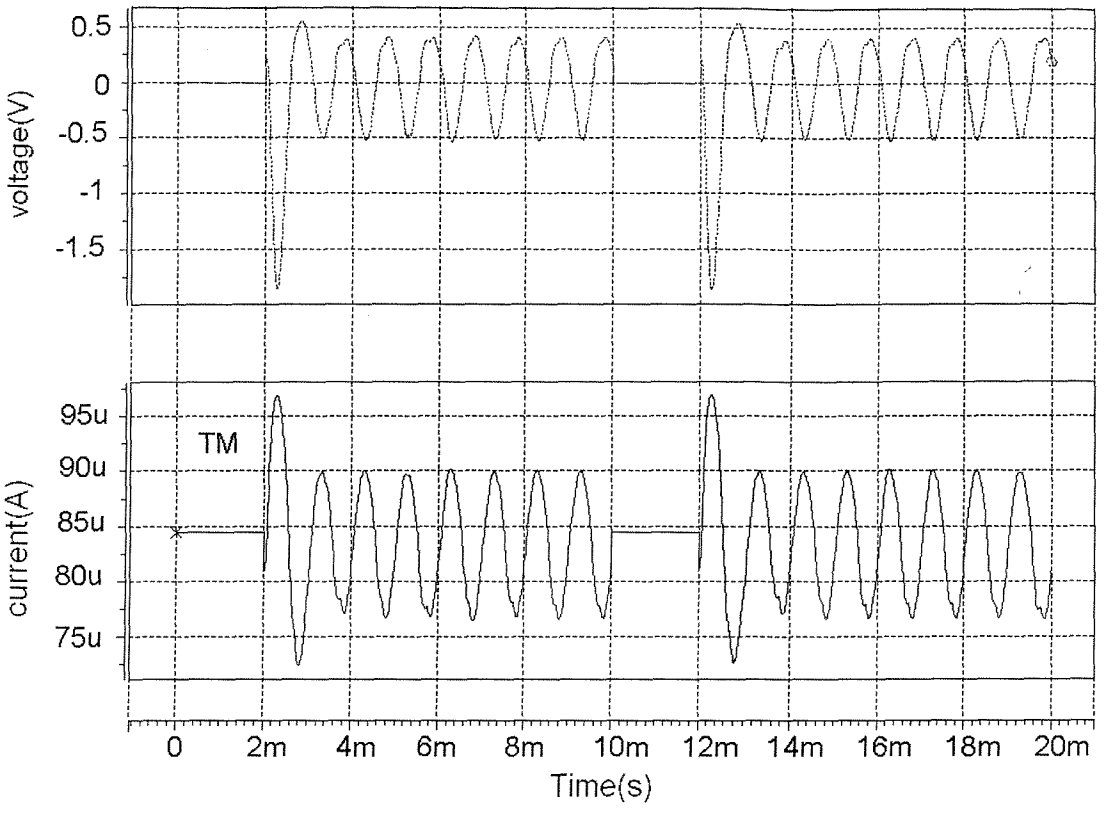


Figure 3-25. Monitored current versus comparator output.

The BICS circuit and the CUT (state-variable filter) were laid out in AMS 0.8μ CYE CMOS technology. (Cadence Virtuoso Layout editor was used for the layout where the layout was done in a full-custom manner.) The equally-delayed non-overlapping clock generator circuit layout, the switched-capacitor input offset reduced comparator circuit layout and the proposed BICS circuit layout are given in Figure 3-26, Figure 3-27, and Figure 3-28 respectively. Figure 3-29 shows the layout of the BICS circuit and the CUT together.

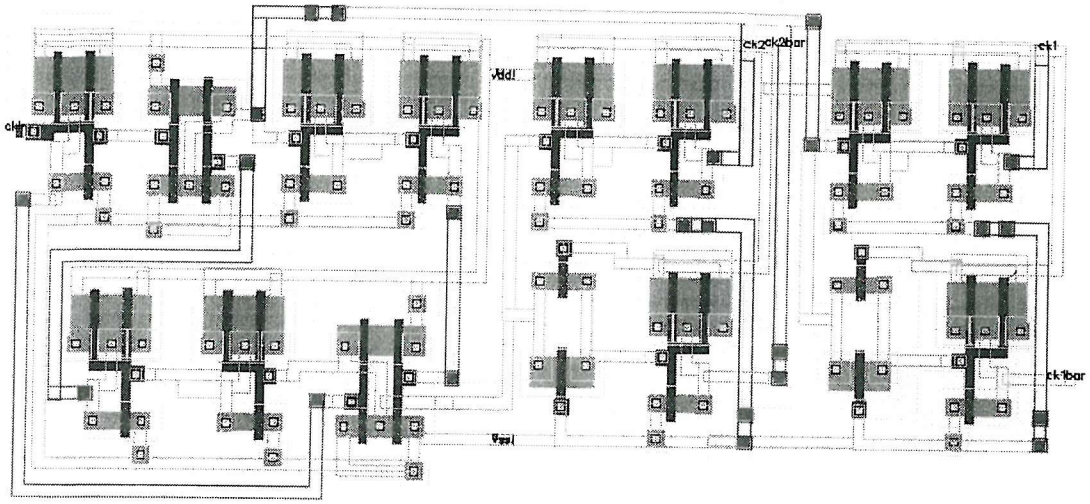


Figure 3-26. Equally-delayed non-overlapping clock generator circuit layout used in the proposed BICS circuit.

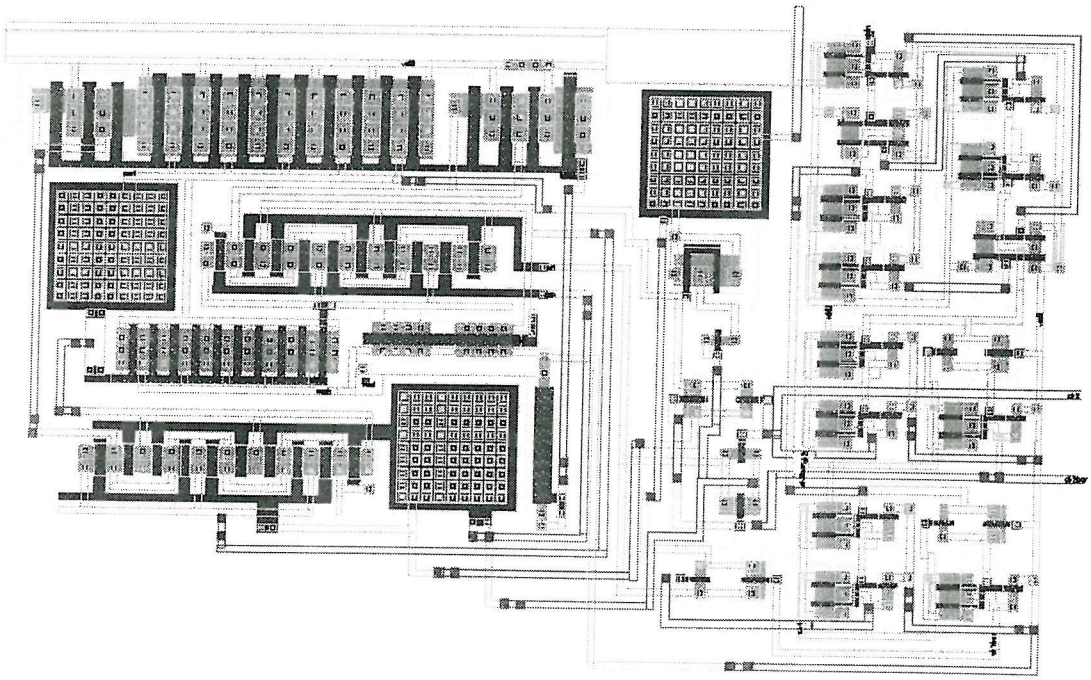


Figure 3-27. Switched-capacitor input offset reduced comparator used in the proposed BICS circuit.

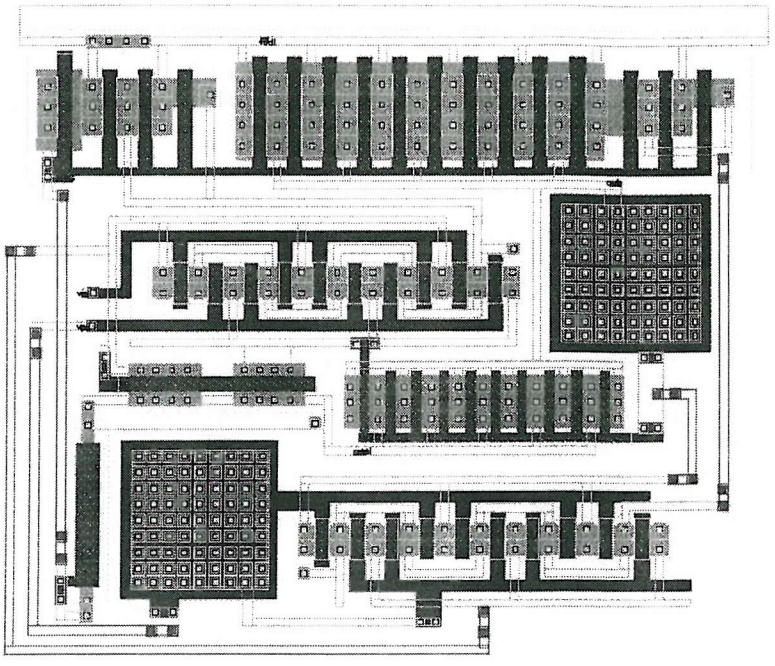


Figure 3-28. Proposed BICS circuit layout.

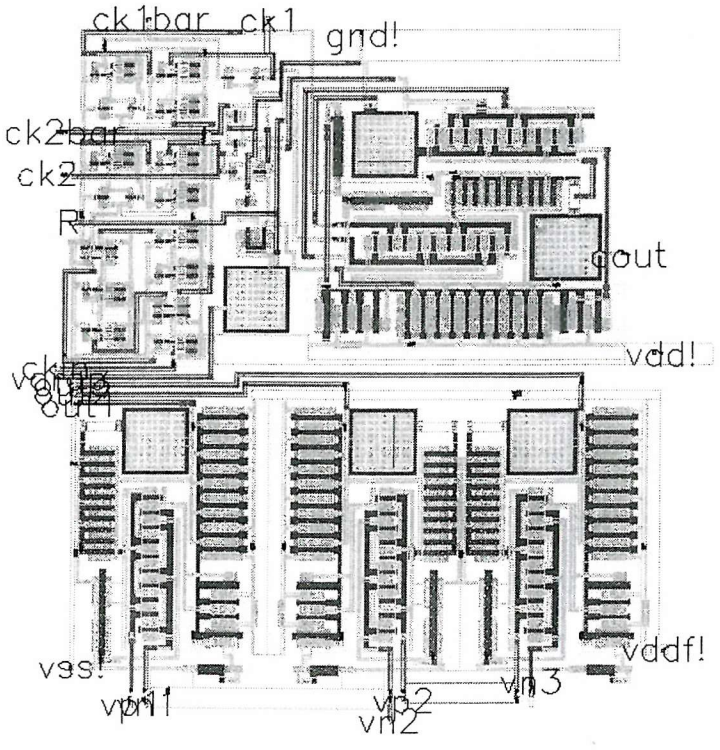


Figure 3-29. Layout view of the BICS circuit along with the CUT filter.

The area of the BICS circuit given in Figure 3-28 is 0.019mm^2 with AMS 0.8 CYE CMOS technology.

One way to measure the correct functionality of the BICS on the fabricated IC is to measure the output voltage of the comparator instead of measuring the current on the shunt element. Therefore in Figure 3-30, Figure 3-31 and Figure 3-32 the post-layout simulation results are given where the output of the comparator is compared with the supply voltage of the CUT. Figure 3-30, Figure 3-31 and Figure 3-32 represent the comparator output voltage versus the CUT supply voltage for the WS, TM and WP process parameter sets respectively.

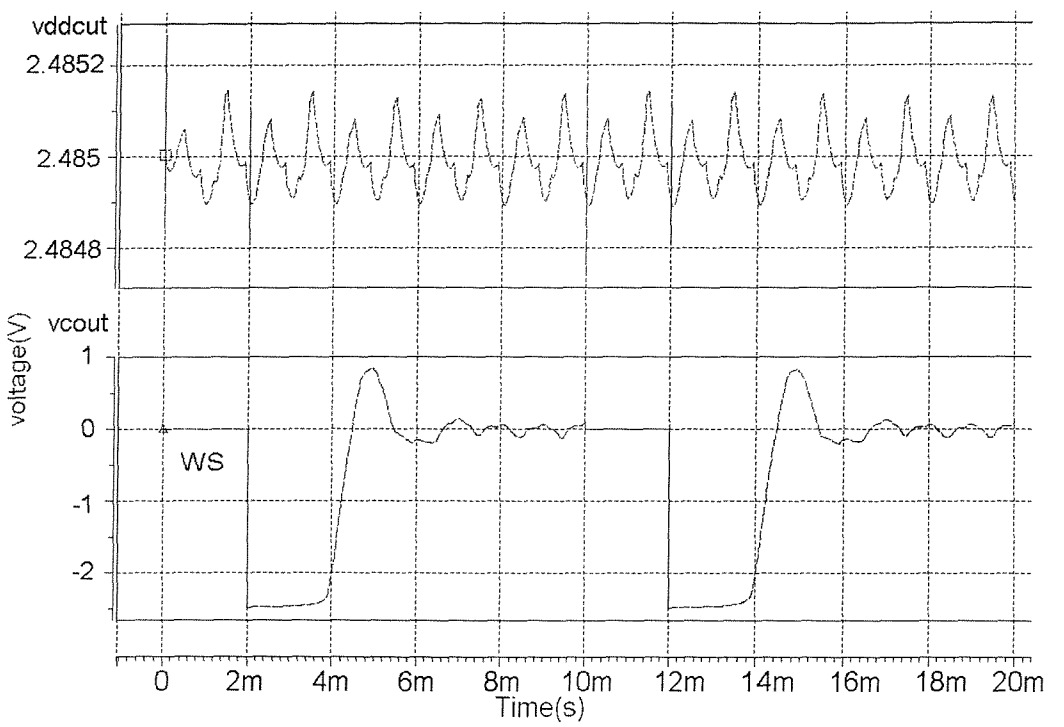


Figure 3-30. Comparator output versus the CUT supply voltage for WS parameter set.

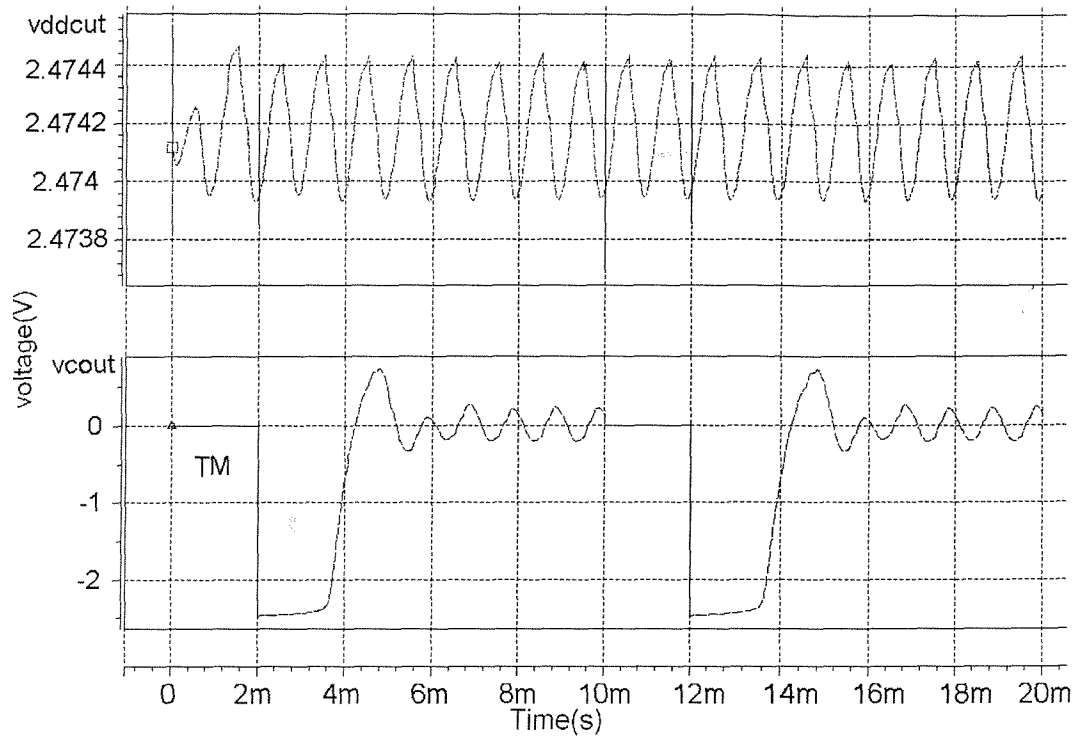


Figure 3-31. Comparator output versus the CUT supply voltage for TM parameter set.

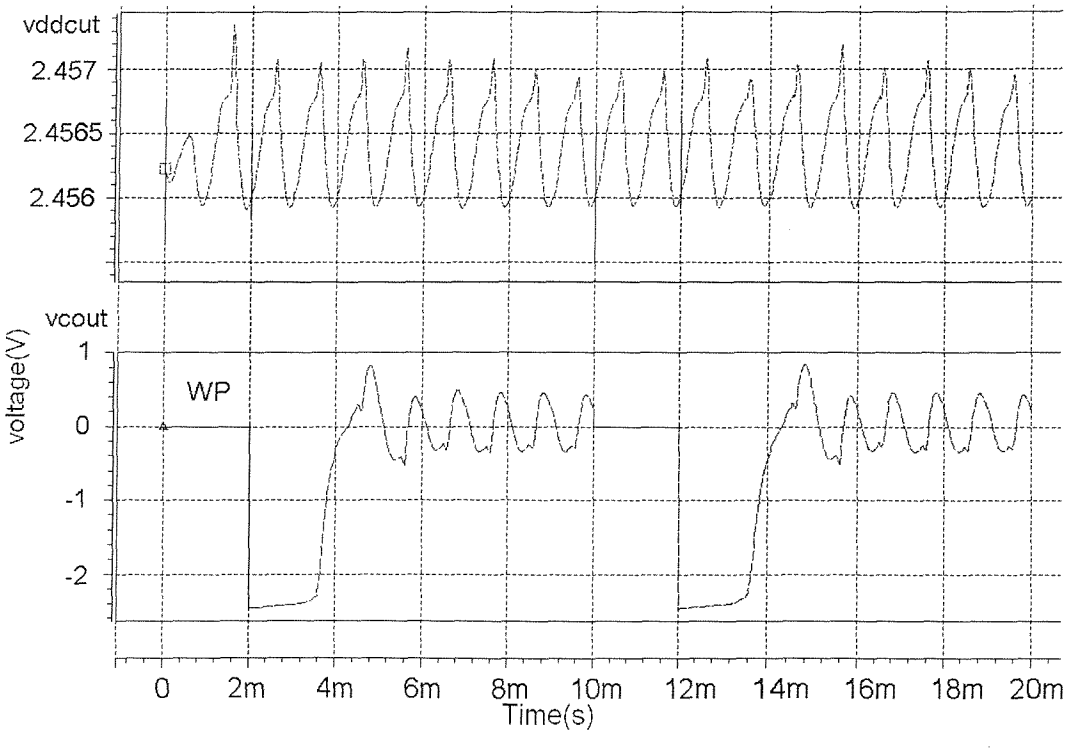


Figure 3-32. Comparator output versus the CUT supply voltage for WP parameter set.

As the comparator has a high gain (63.5 dB), a very small delay will be translated into a saturated voltage at the output of the comparator due to the limited supply rails. As can be seen from Figure 3-30, Figure 3-31 and Figure 3-32, the realized BICS is functioning correctly even when the dynamic change in the supply voltage of the CUT is very small. The dynamic CUT supply change for WS parameter set (see Figure 3-30) is around $200\mu V_{peak\text{-}to\text{-}peak}$ where with this input comparator with 63.5 dB gain causes around $0.3V_{peak\text{-}to\text{-}peak}$ change at its output.

3.4.5 Measurement Results

In order to confirm the correct functionality of the BICS, a number of measurements were carried out on the fabricated IC. The similar signal levels to the one used in section 3.4.4 were applied to the CUT input for the number of packaged ICs containing the BICS and the CUT. Measurement results were obtained using TDS 220 Digital Storage Oscilloscope [74]. Two of obtained results are given in Figure 3-33, and Figure 3-34. As can be seen from the figures, it was not possible to obtain the expected dynamic variation in the CUT supply voltage due to the unwanted noise coming from the measurement environment. The measured comparator output voltage, however, was found to be somewhat similar to the expected ones found by simulation. Note that the dynamic change measured at the output of the comparator given in Figure 3-33 and Figure 3-34 are little bit different as they are for two different packaged IC.

The process variation independent BICS circuit proposed in this chapter is used for the purpose of analogue fault simulation in Chapter 5, where it has been shown how a fault is detected by monitoring the dynamic CUT current with BICS.

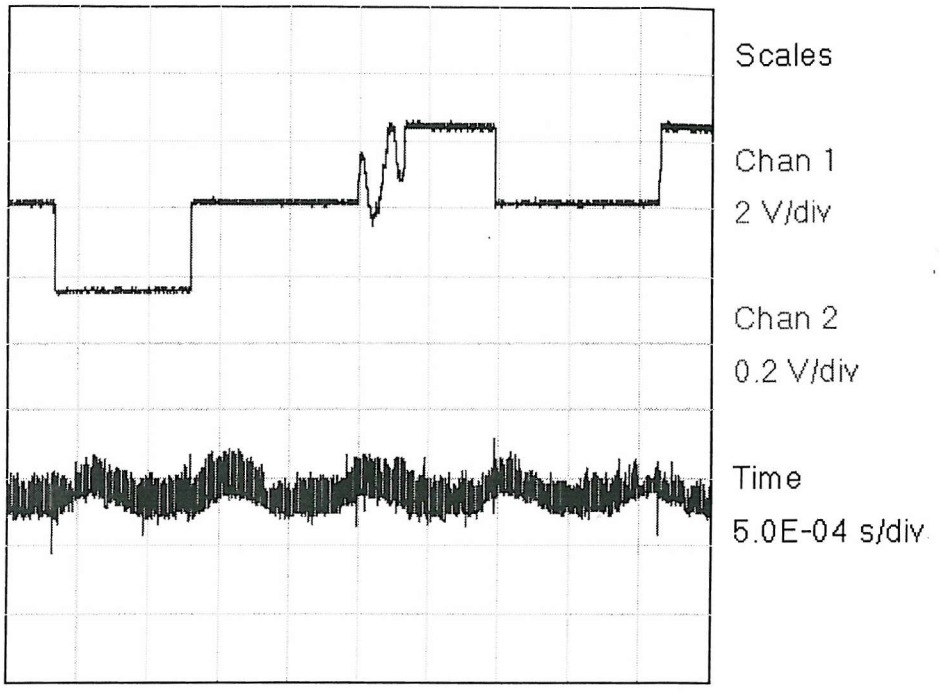


Figure 3-33. Comparator output voltage (top) versus the CUT supply voltage result obtained from the fabricated IC.

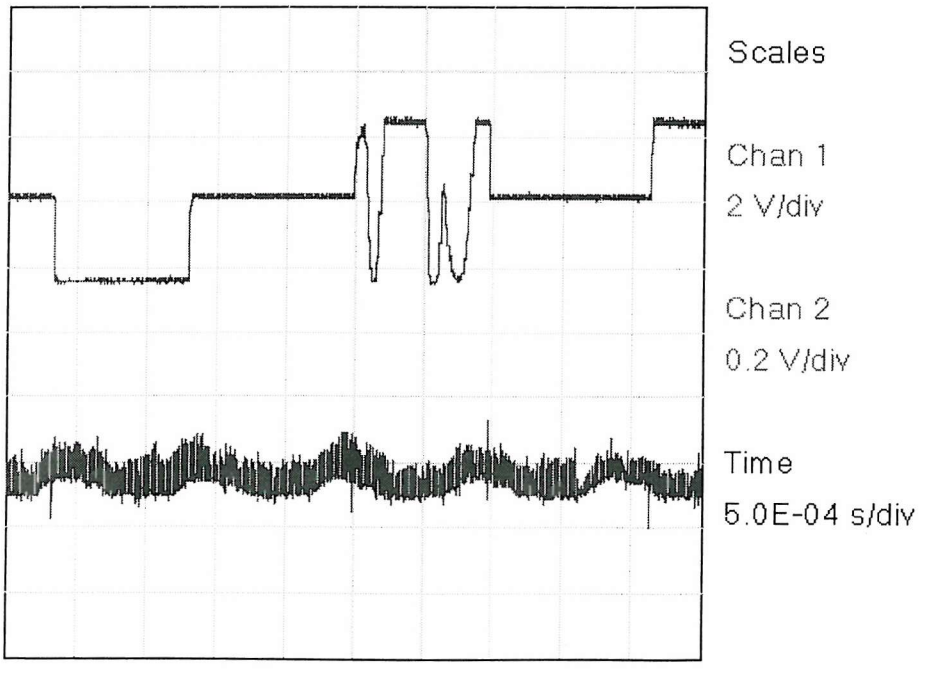


Figure 3-34. Comparator output voltage (top) versus the CUT supply voltage result obtained from the fabricated IC.

4 TESTING ANALOGUE CIRCUITS BY SUPPLY VOLTAGE VARIATION AND SUPPLY CURRENT MONITORING

4.1 Introduction

Test development for analogue circuits has been very much based on the functional and performance specifications [18]. Structural fault modelling for digital circuits, on the other hand, is now well investigated [1] where standard DfT and BIST methodologies for digital circuits are well established. Although there has been some research on the applicability of DfT [49]-[51] and BIST [54]-[57] to analogue circuits, analogue DfT and BIST are still very much done on an ad-hoc basis since there still is not a standard fault model for analogue circuits.

For digital circuits, the structural level of fault modelling is mainly used for the test vector generation [18]. A fault is then said to be detectable when for at least one of the test vectors the faulty circuit behaviour differs from the fault-free one. Analogue circuits, however, do not have that binary distinction which complicates the definition of a fault and the fault detection. Therefore analogue circuit test development has mostly been based on the functional and performance specifications. Consequently, a test developed on the basis of the functional and performance specifications can neither guarantee a certain fault cover nor the

expected quality and reliability levels. Moreover, to test all the specifications of a state-of-the-art large and complex analogue and/or mixed-signal circuit would be time consuming and very expensive. Therefore, the motivation of the work presented in this chapter is to investigate a test generation technique for analogue circuits and its application to a large analogue circuit.

4.2 Marginal Voltage Screening for Digital Circuits

Variable power supply is used as a test technique for digital circuits where it is used for the *marginal voltage* screening [75]-[77]. *Marginal voltage* is defined as the minimum value of the supply voltage that is necessary to just maintain the correct operation of a circuit during a functional test [75]. In addition to the conventional functional test, the upper and the lower limits of the supply voltage (marginal voltages) outside which the circuit fails to operate correctly during the test can be measured. In practice, however, the upper voltage limit is not investigated if the circuit damage is likely to result from operation at voltages in excess of those normally specified. Therefore, in practice the lower marginal voltage is used where for each functional (input) test vector for a digital circuit a separate lower marginal voltage measurement is taken [75]-[77].

The justification of the marginal voltage technique is based on the intuitive assumption that devices which have a marginal voltage substantially different from the rest of the batch may be detected as faulty and should therefore be rejected [77].

Depending on the circuit type, the basic technique of marginal voltage testing can be implemented in three different ways: immediate, permanent, and dynamic

marginal voltage. The first one is used for combinational digital circuits where latter two are mainly used for sequential digital circuits [76].

- ***Immediate marginal voltage***

This type of marginal voltage measurement is carried out for combinational logic circuits where the supply voltage of the CUT is progressively reduced while the input test vector is held constant during the functional test. The immediate marginal supply voltage corresponding to each input vector is then marked as the reduced supply voltage at which the corresponding output vector becomes incorrect. The term *immediate* is used since the output of the combinational circuit changes immediately once the supply voltage is lowered.

- ***Permanent marginal voltage***

Before starting test of sequential circuits the first few input vectors are used in order to set the logic states stored within the circuit to known values. This process is called *initialisation* [76]. Subsequent input vectors then result in output vectors that should follow a known sequence for a circuit that is operating correctly (or sometimes referred to as the fault-free circuit).

In case of permanent marginal voltage analysis, after an initialisation sequence is applied to the circuit, the supply voltage is reduced to a value, say V_{test}, for a short period of time during the input vector "n" of a test vector sequence, while input vector is held constant. The supply voltage is then returned to its normal value. The output vector for $(n + 1)$th input vector and subsequent output vectors to the end of the test vector sequence are then monitored to find the relationship between V_{test} and the marginal voltage for the input vector "n". If subsequent output vectors are the same as the expected values of the fault-free case then V_{test} is above the marginal

voltage. If one or more subsequent output vectors differ from the fault-free functional test results then V_{test} is below the marginal voltage. Therefore, by repeating the test sequence for different values of V_{test} for each test vector the marginal voltage for that test vector can be measured. The term *permanent* is used for this case since the transition of the supply voltage to V_{test} produces stable or permanent changes in the stored logic states of the circuit, which may be detected at any future stage of the test sequence.

- ***Dynamic marginal voltage***

Dynamic marginal voltage testing case is the same as the permanent marginal voltage testing case except the power supply voltage is reduced while the input vector change between successive steps of the functional test. The marginal voltage is the result of a measurement that corresponds to the critical power supply voltage that is required for the circuit to change from one state to another and hence the term *dynamic*.

As stated in [18], the main drawback with these implementations is the resulting test execution time. The normal test vectors have to be applied to the device for various values of the lowered supply voltage. As it can be time consuming to change the power supply of a device under test on a production tester, another implementation is needed.

An alternative solution to these implementations was described in [78] for digital CMOS ICs. In their approach the authors determine the marginal voltage level before the tests are started by means of simulations. Since the allowable process variations can cause the marginal voltage to lie in a certain range, the maximum value of this range was chosen as the marginal voltage to be used later. For the specified vector set only one supply voltage was used, which was a small

threshold higher than the marginal voltage to make the measurement robust. If one of the devices gives a functional failure at this supply voltage level, this indicates that the electrical behaviour has been changed significantly as a result of the occurrence of a defect. This technique requires some simulation time before test, but the test execution time is reduced considerably [18].

4.3 Variable Supply Voltage Technique for Analogue Circuits

One way to achieve some control over the behaviour of transistors within an analogue circuit is varying the supply voltage in conjunction with the inputs. Bruls used this idea to test a class AB amplifier at various supply voltage levels [18]. He used the inductive fault analysis (IFA) technique to insert processing defects into the layout of the IC in a random manner. A'ain, et al applied a ramped power supply voltage to test simple 2-stage operational amplifier circuits, and for exposing floating gate defects in analogue CMOS circuits [19], [20]. The same authors applied an AC supply voltage to analogue CMOS circuits [21]. They achieved high fault covers with these tests, although the sizes of the circuits and the numbers of faults were small.

In this chapter, it has been shown how varying the supply voltage of an analogue circuit block can increase the fault cover of that block. Unlike previously described works in [18]-[21], a larger circuit element –a phase locked loop– was used as a test vehicle where only structural short circuit faults were taken into account. The technique is based on using structural fault models where it is targeted to reduce expensive testing costs due to specification and performance based tests by considering testing in a structural manner before the production of the first silicon.

4.4 CMOS PLL for the CUT

A CMOS Phase-Locked Loop, Figure 4-1, was used to test the application of the technique mentioned in the previous section to a larger circuit. The circuit was designed using the MIETEC 2.4μm CMOS technology. The PLL circuit operates with ±5 V supply voltage. As the PLL circuit was designed elsewhere [79], here details about this design will not be given.

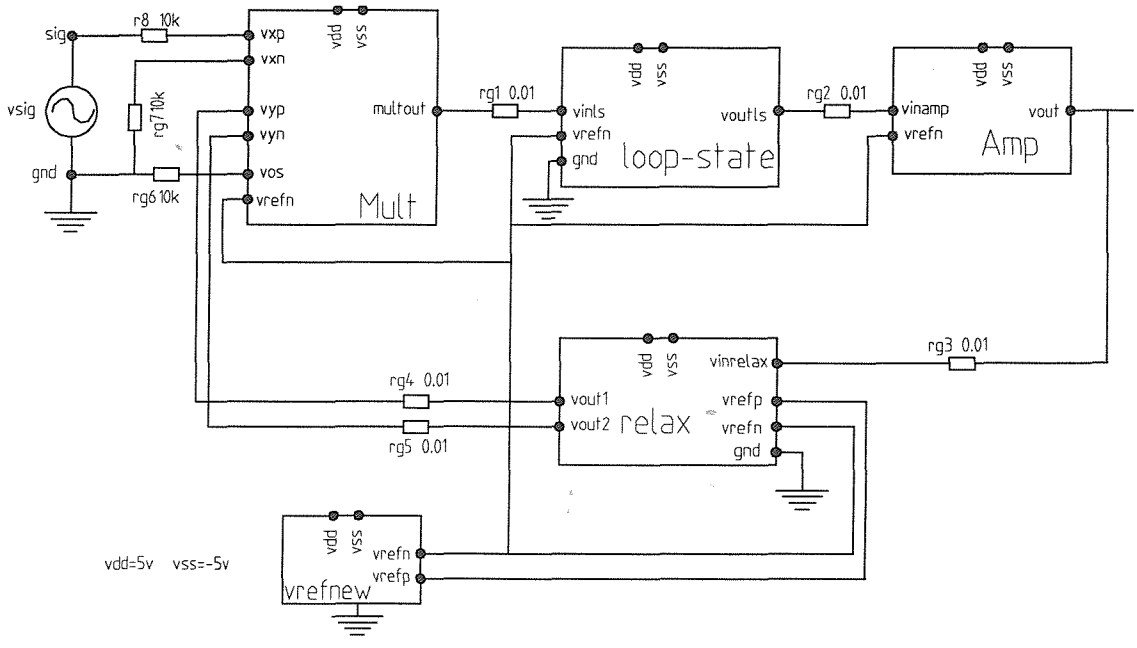


Figure 4-1 CMOS Phased-Locked Loop [79].

4.5 Fault list generation

As mentioned above, in this chapter only structural bridging/short circuit faults for MOS transistors were considered while investigating the variable supply voltage technique through the use of circuit given in Figure 4-1.

The repeated insertion of circuit faults by hand into a SPICE netlist in order to carry out simulations is tedious. Therefore the ANTICS fault simulator [80] was used to inject faults into the SPICE netlist and to analyse the simulation results. Gate-to-source and gate-to-drain short fault models for MOS transistors as shown in Figure 4-2 only were used. This is distinct from the open-gate fault model used in previously reported work [20].

Short faults were modelled with a resistance, Rs in Figure 4-2. To simplify matters, Rs value was chosen to be 10Ω.

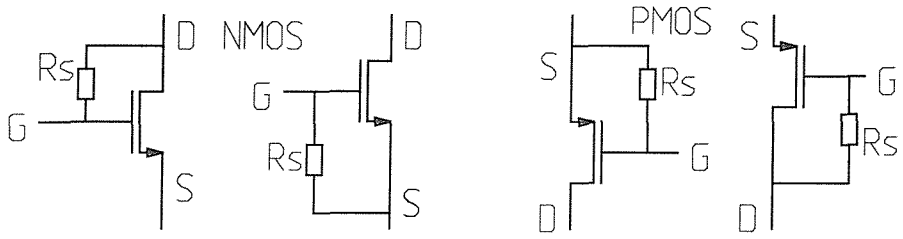


Figure 4-2. Structural short fault models for NMOS and PMOS transistors.

4.6 Fault Simulation and Fault Coverage

The total number of injected faults was 190, of which 28 were redundant (the short circuits already existed as part of the circuit configuration) and 33 were equivalent (the same inter-nodal short was injected at two separate transistors) hence, 129 distinct faults were simulated. Fault simulations were done using HSPICE in a serial manner, where each faulty circuit along with the fault-free one were simulated one after another and results were saved for fault coverage purposes.

Monte Carlo simulations were performed for each of these 129 faults and the RMS value of the AC component of the supply current and the DC supply current were measured. The PLL was simulated as a whole where the input stimulus was chosen to be a sinusoidal signal within the locking frequency range.

The fault coverage was then evaluated such that a fault was considered detectable if the 3σ points (see Figure 4-3) of the faulty and fault-free current distributions did not overlap. This separation or gap between two distributions is defined in [81] as:

$$gap = (\mu_f - 3\sigma_f) - (\mu + 3\sigma) \qquad (4\text{-}1)$$

where μ_f is the mean value of the faulty circuit response and μ is the mean value of the fault-free circuit response. For a fault to be detectable the gap shown in Figure 4-3 must be greater than zero.

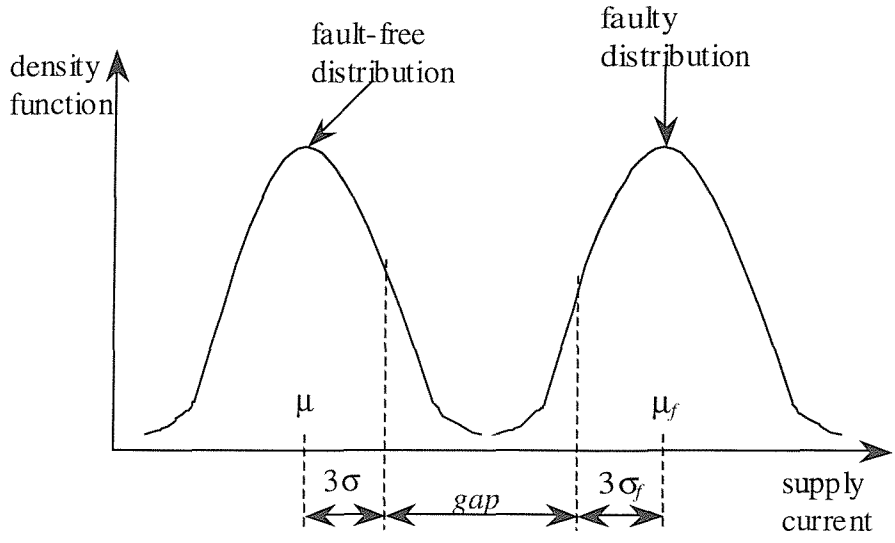


Figure 4-3. Probability distribution function of fault-free and faulty supply currents proposed in [81] for calculation of the *gap* given in (4-1).

The fault coverage was evaluated at different supply voltages using the DC and AC RMS values individually and combined. In the remainder of this chapter fault coverage is presented for different supply voltage levels where both positive and negative supplies are considered to vary in order to increase the fault coverage.

4.7 VDD Change for whole PLL

Table 4-I shows the fault coverage for the PLL as the supply voltage was varied between 4.0V and 5.3V. As can be seen from Table 4-I, the fault coverage changes very little for different supply voltages. The three figures given for each supply voltage value are the fault coverage found by measuring the RMS value of the AC component of the supply current; the DC supply current and the cumulative total of the two measures. For each supply voltage level and each test used (RMS or DC) a different set of faults is detected. This means that for instance some of the faults that are not detected by the RMS test using a 5V supply voltage, might be detected by DC test or by RMS test at another supply voltage level. Therefore, it is not correct if one compares the figures given in the table among themselves. A number of faults are detected by both measurements, which is why the sum is only around 10% higher than the RMS value alone.

Table 4-I. PLL Fault Coverage for varying supply voltage

VDD [V]	Fault Coverage [%] RMS	Fault Coverage [%] DC	Fault Coverage [%] RMS + DC
5.3	67	38	74
5.0	64	40	73
4.8	67	39	77
4.7	69	39	77
4.6	68	44	77
4.5	67	40	77
4.0	65	41	76

Table 4-II shows the coverage obtained by combining two or more of the tests from Table 4-I. All three given figures are fractionally above those of Table 4-I. This means that a large common subset of faults is detected by all the tests, with a small number of extra faults detected by each test individually.

Table 4-II. PLL Fault Coverage for combined tests

Multiple VDD [V]	Fault Coverage [%] RMS + DC
5 + 4.7 + 4.6	79
5.3 + 4.8 + 4.7 + 4.6	80
4.6 + 4.7	78

In other words, it appears that very few faults are sensitised, with respect to the supply current, by varying the supply voltage of the PLL as a whole. Therefore it is investigated how one can increase the total fault coverage by varying the supply voltage for one block within the PLL, while keeping the supply voltage level of other blocks at the normal operating voltage level. The idea is that if most of the undetected faults are within one block of the whole circuit then varying the supply voltage for the entire circuit might not sensitise some of the faults within this block. Next section discusses how the total fault coverage can be increased by changing the supply voltage for the VCO block within the PLL circuit.

4.8 Change in VDD of one block

Most of the undetected faults occurred within the VCO (Voltage Controlled Oscillator) block, (relax in Figure 4-1) therefore the exercise was repeated by varying the positive supply voltage of the VCO block only while keeping the positive supply voltage of all other blocks at 5V. It was found that with the supply

voltage of the VCO block being changed to 4.5V while the supply voltage of the other blocks was held at 5.0V gave the best fault coverage. This proves the idea that varying the supply voltage at sub-block level for a complex circuit increases the fault coverage is valid.

The combined fault coverages of this test and the previous tests are shown in Table 4-III. The DC fault coverage quoted here is that obtained using the first VDD value of the row in the table. The cumulative totals for the single DC test and the two or three AC supply current tests are given.

Table 4-III Cumulative Fault Coverage with VDD varied for one block.

Multiple VDD [V]	**Fault Coverage [%] RMS + DC**
5 + 4.7 + 4.6 + 4.5(VCO)	81
5.3 + 4.8 + 4.7 + 4.6 + 4.5(VCO)	82
4.6 + 4.7 + 4.5(VCO)	80
5 + 4.6 + 4.5(VCO)	83

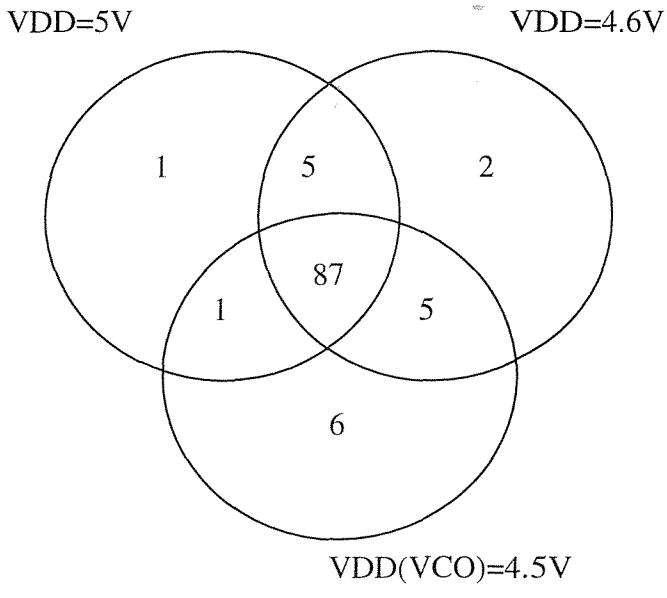


Figure 4-4 Venn diagram of fault coverages of different tests.

Figure 4-4 shows a Venn diagram for the fourth test in Table 4-III. Again, it can be seen that most faults are detectable by all three tests. Six extra faults are detected by changing only the supply voltage of the VCO. While this only increases the overall fault coverage by around 4%, these faults would not otherwise be detectable.

4.9 VSS Change

Even by varying the positive supply voltage for just the VCO block, a number of faults in this block remained undetected. How about if one varies the negative supply voltage of the VCO block? To investigate this, the negative supply voltage (VSS) of the VCO block was varied while keeping other blocks to work at ±5 V supply level (normal operating condition). Now, for VSS=-4.7V (instead of -5.0V) the fault coverage obtained by RMS test was 66% compared with a fault coverage of 70% when VDD=4.7V. Therefore, monitoring the current while changing VDD seems to be better in terms of fault coverage. Note that in the case of varying VDD the supply current is measured from the positive supply (Idd), in the case of varying VSS the supply current is measured from the negative supply (Iss).

For various VSS values (-4.6V, -4.65V, -4.7V, -4.75V, -4.8V) of the VCO block it was still not possible to detect any of those faults that could not be detected by varying the positive supply voltage of the VCO block.

4.10 Undetected Faults

The fault coverage using the DC and AC supply current tests was increased from 73% to 83% by using three different supply voltages. Nevertheless, 17% or 22

faults remain undetected. The reason why these faults remain undetected will be discussed in this section. 13 of the 22 undetected faults occur in the VCO block. Figure 4-5 shows the circuit diagram of the VCO. Of these 13 faults, 8 occur in the voltage divider chains on the right hand side of the circuit (in Figure 4-5). These undetectable faults include the gate-source shorts on M26, M27 and M28. Although these faults would affect the functionality of the circuit, the effect in terms of the supply current would be negligible, unless they were to force the PLL as a whole to cease to function. Therefore it is not surprising that a supply current test is unable to find them. More significantly, it should be noted that the original premise -that stimuli that would cause transistors to change their region of operation are ideal for supply current testing– cannot be satisfied for such circuit configurations. M26 and M27 are connected so as to be in permanent saturation.

Other undetectable faults include the gate to source shorts of M18 and M21 in Figure 4-5. Again, from the circuit configuration, it would be almost impossible to apply any stimulus that would cause these transistors to switch their mode of operation as transistors M18 and M21 are deeply embedded within the circuit. It is reasonable to suppose that this test technique cannot provide a significantly higher fault coverage than that found here for voltage-mode circuits operating with ±5V supplies.

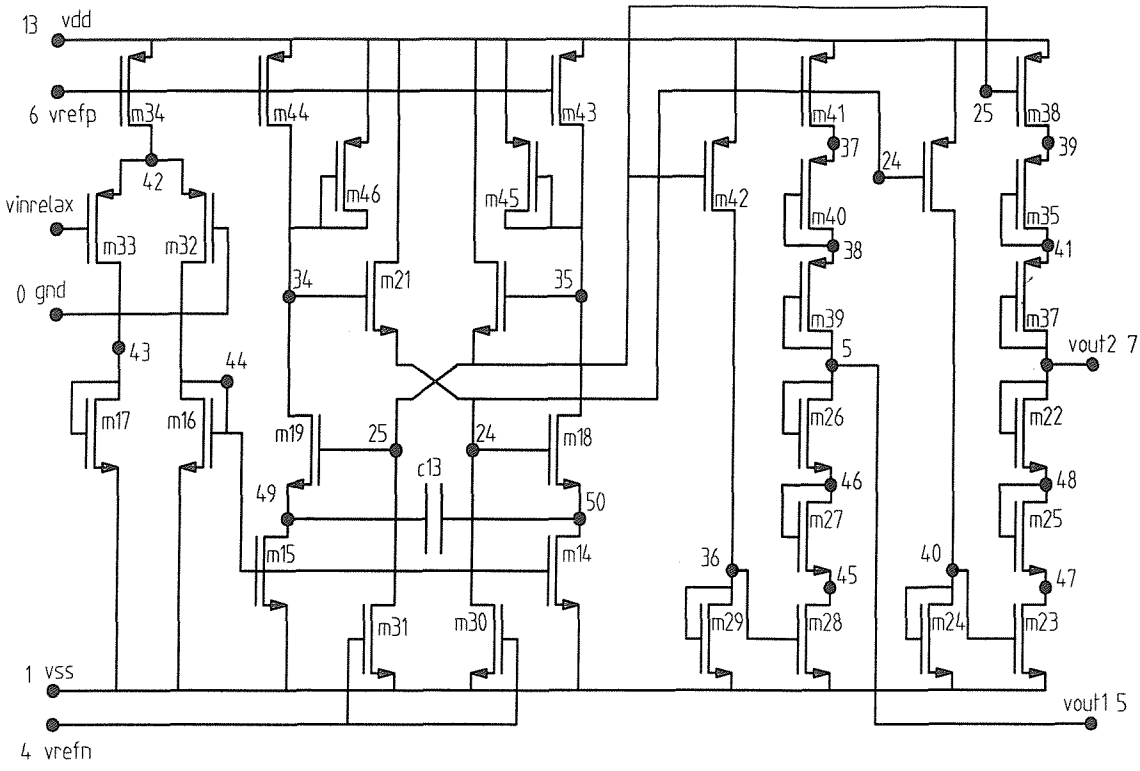


Figure 4-5 Voltage Controlled Oscillator [79].

4.11 Conclusions

The technique that uses variable supply voltage for test has been shown to increase the fault coverage of a complex analogue CMOS circuit by 10% to 83%. Clearly this increase in the fault coverage is not very promising. It can be assumed that this technique is not very efficient with the circuit topologies discussed above, especially when there are number of transistors are connected in a cascode manner between the supply rails within the circuit, as is the case in Figure 4-5.

With the transistor feature sizes getting smaller and smaller with newer technologies, the supply voltage levels also keep decreasing [14]. This will result in reduced number of cascode connected transistors as the threshold voltage levels are not shrinking linearly for the CMOS transistors with the feature size reduction [70],

[72]. The variable supply voltage test technique, therefore, might prove better for today's deep sub-micron designs.

5 ANALOGUE FAULT SIMULATION

5.1 Circuit Simulation

The more complex the design is, the greater the chance that some parts of the system will not function properly together or function in all cases. Simulation is used to analyse a system before the production process to see if the system matches the desired specifications. The two most important reasons why simulation is used before manufacturing the actual device are; it is safer to verify the correct operation of the circuit by means of simulation and secondly simulation is cheaper than verification of the correct operation after the manufacture.

In the rest of this chapter, unless otherwise stated, by *circuit simulation* transistor level simulation for analogue/mixed-signal circuits is meant. The accuracy and speed of the simulation depend on the accuracy of the simulation algorithms and the accuracy of the device models used within the simulator. General circuit simulation algorithms, as used in SPICE and SPICE-like simulators are well documented [82], [83]. In the transistor level circuit simulation process, the set of non-linear, first-order simultaneous differential and algebraic equations (DAEs) given in (5-1) must be solved:

$$\mathbf{f}(\boldsymbol{x},\ x',\ t)=0 \qquad (5\text{-}1)$$

where **f** is a vector of expressions, x is the vector of circuit variables, x' is the derivative of x with respect to time and t represents time.

DAEs have been investigated extensively by numerical mathematicians [82]-[85]. For SPICE-like simulators in order to solve DAEs; numerical integration is used to discretise the time, the Newton-Raphson (N-R) method is used to linearise the non-linear equations and LU factorisation is used to solve the matrix equations [87].

Generally speaking, a circuit simulator consists of three parts: the input or network description part; the simulator or the network analysis part; and the output or postprocessor part [82]. Circuit simulators spend most of their time either evaluating non-linear device models or solving the linearised system of circuit equations. Model evaluation time dominates for small circuits, whereas equation solution time dominates for large circuits [88].

The input part of the simulator is used to describe a circuit, to describe the excitation and to control the analysis. There are many ways to describe the circuit. Traditionally the circuit has been described in a text file where mnemonics (or full words) are used in order to form records describing one element, signal or command. A parser reads this file to check for syntax rule violations. After that the file is compiled and the data structures needed for the analysis are created. Other forms of data inputs include physical layout entry, where integrated circuit or printed circuit board layouts are converted into a textual description and schematic entry, where a library of element-symbols is used in a schematic editor to draw the circuit [82].

The simulation part of the simulator performs the main computational task. This task can be DC operating point analysis, transient analysis, frequency domain analysis, sensitivity analysis, noise analysis, statistical tolerance analysis etc. [82].

The output part of the simulator takes the results of the circuit analysis obtained by the simulator and allows us to be able to monitor those results in textual or graphical manner [82].

In this thesis, the discussion about circuit simulation is limited to the simulation (or sometimes referred to as core) part of the simulator only. The structure of the rest of the chapter is as follows. First non-linear DC and transient analyses for transistor level simulation are briefly discussed. Later, digital simulation is explained. After that, mixed-mode simulation for mixed-signal circuits is dealt with in detail. Finally, fault simulation with the main emphasis on analogue CMOS circuits is investigated.

5.2 Non-linear DC Analysis

Non-linear DC analysis is the problem of solving a system of non-linear equations describing an electronic circuit, which consists of non-linear devices [82]. There are different situations where such an analysis is important. Mostly, non-linear DC analysis is performed to find out the values of voltages and currents in an electronic circuit with the power supply switched on, but with no excitation at the input. This information can then be used as the initial condition for a transient analysis of the same circuit.

Systems of non-linear equations cannot be solved analytically except for trivial examples. They require iterative methods. Both equation formulation and equation solution depend on how the non-linear element characteristics (or models) are expressed. There are methods of describing the models. Usually, the model is described by non-linear analytical functions with continuous derivatives produced

either by fitting curves to empirical data or by analysis of the physical phenomena in the component [82]. The N-R method is then used for solving the non-linear equations. In this method, the non-linear system of circuit equations is transformed into a linear system, which changes at each iteration. In other words, the problem is converted into a successive formulation and solution of a system of linear equations.

There is one more matter to be considered when discussing non-linear DC analysis [82]: the characterisation (i.e. modelling) of the non-linear components. Since it is not intended as a subject matter of this project, for further details about the characterisation of the components refer to [82].

5.3 Transient Analysis

Here, by the transient analysis the analysis of electronic circuits in the time domain is meant. Time domain circuit analysis without use of a computer is effectively restricted to simple linear circuits with no more than three dynamic elements. Numerical methods are needed in order to cope with the solution of non-linear algebraic and ordinary differential equations that model non-linear resistive and dynamic electronic elements. The main technique used is discretisation of time and successive computation of the response value from one time instant to the next [82].

The main disadvantage of using numerical methods is the excessive computer time needed to compute the response over a long time interval. The elapsed computer time can get very large as the circuit gets more and more complex. In addition, as the response is computed at discrete points in time, it remains unknown in between these points.

After the development of the well-known circuit level simulator, SPICE, at University of Berkeley [89] there have been many different variants of SPICE simulators. The simplified transient analysis algorithm implemented in SPICE-like simulators is given in Figure 5-1. Modified Nodal Analysis (MNA) is a method, which transforms the circuit topology into a set of non-linear differential equations [82], [83]. The unknowns of these equations are node voltages or branch currents of the circuit being simulated.

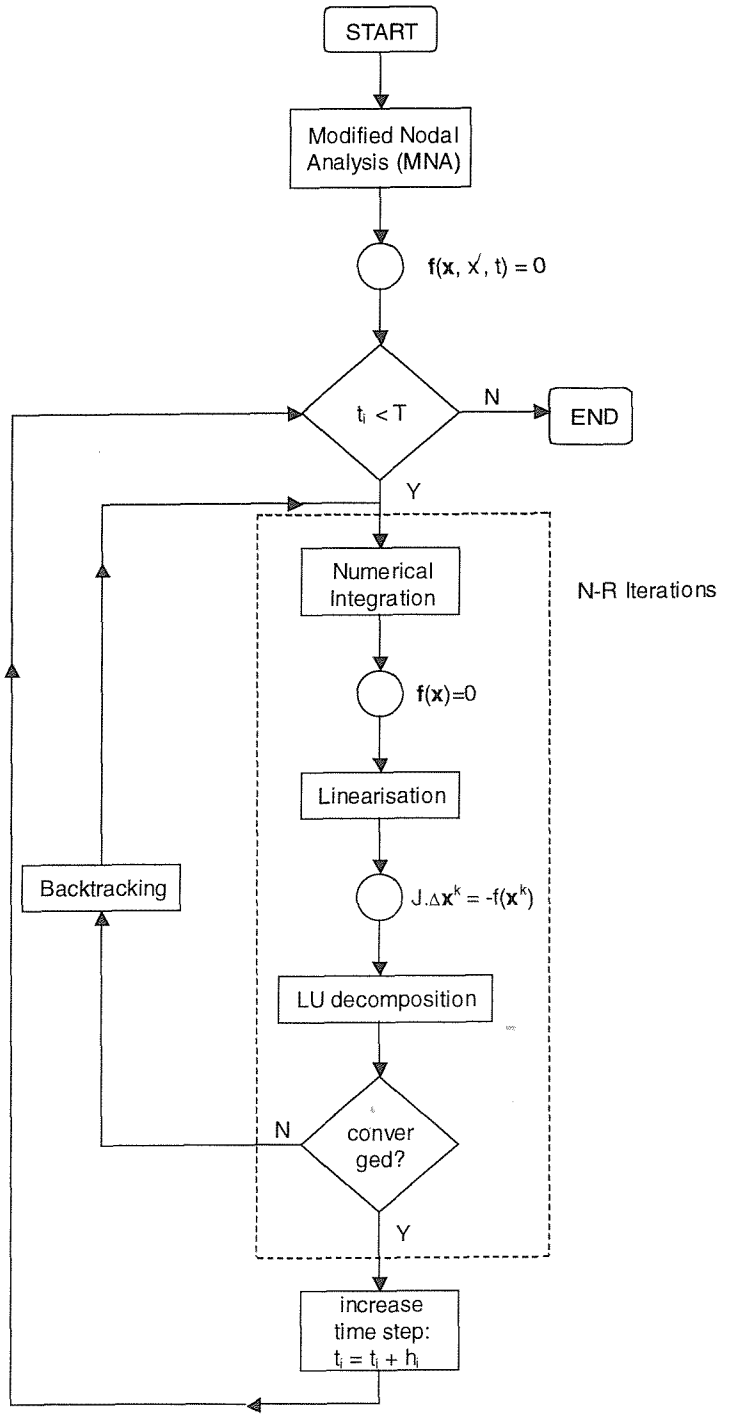


Figure 5-1. Flow-chart for transient analysis algorithm used in SPICE [4].

5.4 Logic (or digital) Simulation

In analysing digital circuits, the main concern is that signals reach a correct Boolean value at a particular time, not with the exact behaviour of individual transistors. That means one can model such circuits with less detail than analogue circuits.

There are a few approaches to simulating logic circuits such as *compiled simulation* and *interpreted simulation*. In the first approach, the description of the circuit is linked in directly to the simulation algorithms. This method can be fast, as an efficient executable program is generated for each circuit. The interpreted simulation approach takes the circuit description and generates a data structure, which is updated for each time step in the simulation. As today's standard hardware description languages such as VHDL and Verilog are based on compiled simulation technique, in the next section compiled simulation is given in more detail.

5.4.1 Compiled Simulation

Let us take the circuit given in Figure 5-2 as an example. This circuit can be represented by the C code given in Figure 5-3. This code can be complied (provided that the appropriate header files are provided), and linked with code to evaluate the logic functions. Inputs to the circuit could be taken from a file or generated by another piece of code. This model suffers from the disadvantage that the gates do not have any delays associated with them [82]. The code describing the circuit can be evaluated each time an input changes, but the output will change instantaneously. In order to include timing information in the model, it would be necessary to make `signal` into a data structure containing past and present values as well as the time

at which the data changed. This would make the simulator harder to use as the logic functions would have to take extra arguments such as the delay and the output signal.

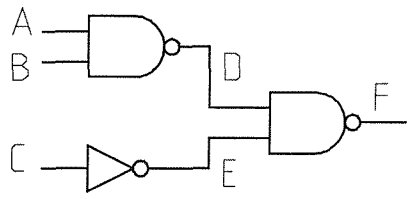


Figure 5-2. Example logic circuit.

```
signal cct (signal a, b, c)
{
      signal d, e, f;
      d = nand (a, b);
      e = inv e;
      f = nand (d, e);
      return f;
}
```

Figure 5-3. C description of circuit given in Figure 5-2.

There is another drawback with the use of a conventional programming language. The description given in Figure 5-3 is ordered: `d` and `e` must be evaluated before they can be used to evaluate `f`. This makes the modelling of even a simple latch difficult. A C description of a latch would need to be repeatedly evaluated until all activity had settled. Hence, the simulator would need some means of monitoring activity and this might well complicate the circuit description still further. The

problem arises because C and other programming languages are sequential, whereas, hardware is essentially parallel. Therefore, a true hardware description language, such as VHDL (Very High Speed Integrated Circuit Hardware Description Language), must be parallel in nature.

5.4.2 Hardware Description Languages

Digital circuits can be modelled in terms of gates. For many digital systems, this level of detail is unnecessary. For today's large digital circuits their complexity makes gate-level simulation impractical. By simulating at a functional or behavioural level, the complexity can be reduced but an important detail can also be lost. In order to simulate a circuit or system at a functional level, it needs to be described in some computer-readable form. While it is possible to describe circuits using conventional programming languages, there are difficulties as discussed before. There are a number of hardware description languages (HDLs) that have been developed to allow arbitrary circuit descriptions to be simulated at different levels of hierarchy, such as RTL (Register Transfer Level) etc.

A general HDL does have to be very different from a programming language. VHDL has been based on Ada programming language, and Verilog has been based on C/C++ programming language. With a HDL one needs two basic features in order to be able to represent the desired hardware. First, the HDL must have concurrent statements to avoid the problem of knowing which statement to evaluate first. Therefore statements in HDLs are generally concurrent unless identified as sequential. Secondly, it must be possible to model timing effects for the accurate simulation.

Delays can be modelled by specifying that a signal changes after a certain time. For instance, in VHDL this can be modelled for an NAND gate as follows:

```
z <= x NAND y AFTER 10 ns;
```

To do the simulation in an efficient manner, it must also be possible to detect when an enabling signal changes, so that a model is not repeatedly simulated even when its outputs would remain unchanged. In VHDL this can be done for a clock edge as follows:

```
WAIT UNTIL rising_edge(clock);
```

HDLs must allow circuits to be described at various levels of abstraction in order to give a user freedom to choose which level of abstraction to choose from. In the simplest case, a digital circuit can be modelled as a list of gates with a HDL. This level of abstraction is generally referred to as *structural* description. The next level of description is the *Register Transfer Level* (RTL) or *dataflow* level. In RTL level, the behaviour of a system is described in terms of signal flows between flip-flops or registers. Both structural and RTL descriptions can be interpreted very easily and used to provide descriptions for simple event-driven simulators. *Behavioural* and *functional* levels of description use programming language type constructs to model systems at a higher level of abstraction. To interpret such descriptions one needs a simulator that can handle the functional and time-dependent behaviour.

More detailed investigation into HDLs mainly for behavioural fault modelling for analogue/mixed-signal circuits is given in the next chapter.

5.5 Mixed-Signal Simulation

Circuit level simulators deal with voltage and current values. These values are continuous, which means that there are an infinite number of possible values. Moreover, since analogue signals are continuous by nature, they cannot instantaneously change to different values. Matrices are created inside the analogue simulator for relating to these signals. Circuit level simulators generally use an integrating algorithm that takes variable steps in time, which is referred to as *time steps*, and solves the matrix for all voltages and currents for each time step.

Logic simulators, on the other hand, use only a finite number of discrete values, which is called *states*. These states change only at specified times, which is called *events*. When a function generates a state, it places that state into an event queue so that other signals attached to that function could evaluate the effect of new state. By limiting the range of state values, solving the circuit at discrete times, and solving for only the functions affected by a changed state, the logic simulator can solve for circuit function faster than a circuit level simulator that could evaluate an equivalent analogue circuit [90].

Due to the reasons stated above, logic simulation perhaps is two orders of magnitude faster than a circuit-level transient analysis of the same circuit [22]. The accuracy, however, of the simulation is much less. This level of accuracy is sufficient for digital circuits. The trend in VLSI circuit design, however, is to put entire mixed analogue and digital systems on one integrated circuit, referred to as System-on-a-Chip (SoC). Not only are rigorous design styles needed but also it is impossible to model and verify such designs by any means other than simulation [82]. If the design contains both analogue and digital parts, a problem created. An analogue design cannot be accurately simulated using a logic simulator. In principle,

it would be possible to take a large mixed-signal design and simulate the entire circuit at transistor level, but the CPU time required would probably be prohibitive, notwithstanding the numerical problems that might well result from simulating a complex digital circuit [82].

One way to cope with this difficulty is to simulate the analogue parts using a circuit-level transient analysis algorithm, while simulating the digital parts with a logic simulator. This will work well for simple circuits.

If the analogue part of the circuit is more complex then there is a clear problem with the technique to simulate analogue and digital parts of the circuit separately using separate simulators. For instance suppose that the digital part of the circuit might produce analogue signals through a D/A converter. These analogue signals might be combined with analogue input signals, in analogue circuitry. The digital circuitry itself might have a loading effect on the analogue part. It is precisely these effects that the designer may wish to test and verify by simulation. Independent simulation of the two parts is therefore not sufficient. A true mixed-signal simulator must be capable of passing signals between two (or more) very different simulators, ensuring that any signal transformations are handled accurately and that any signal changes occur at precisely the right times [82].

The conversion of signals between two simulators and how to ensure that the two simulators remain synchronised are main two problems to be taken into account [82]. Circuit-level simulators use voltages and currents to represent the state of a circuit, while logic simulator use discrete states. Moreover, circuit-level nodes are bi-directional while logic elements are unidirectional. Therefore, the synchronisation problem is perhaps harder. Circuit-level simulators use complex numerical algorithms and therefore comparatively slower than logic simulators.

5.5.1 Signal Conversion

One needs to be able to efficiently convert logic signals to analogue equivalents and vice-versa for the sake of the accuracy of the mixed-mode simulation. The analogue to logic interface can be relatively simple. The logic 1 can be defined when the analogue value is more than a threshold and similarly logic 0 when the analogue value is less than a threshold. These two thresholds can be the same, in which case there are only two logic states. If the thresholds are different then there is a transition state, which is known as unknown state.

A logic state can be converted into an analogue voltage or current. Two problems might occur are; how to translate logic X (unknown state) and logic Z (high impedance state) into analogue voltage and current, and the problem of instantaneous transition of logic signal, which will cause problems in a circuit-level simulator. The time step in a circuit simulator can be adjusted such that rapid changes require shorter time steps than do slow changes. An instantaneous change would eventually cause the time step to be cut to zero and the simulation to fail.

The translation of logic states other than 0 and 1 to analogue values is technology dependent, and to a great extent a modelling decision that is best avoided, if possible [82]. Whatever assumption is made, it cannot be guaranteed to be accurate.

One way of solving the problem of instantaneously changing a voltage value is to associate a capacitance with each output node, which would have the effect of limiting the rate of the change of the voltage. Another method is that the rate of the change of a node voltage can be limited by limiting the rate of the change of the voltage sources or of the resistances. For further details refer to [82].

5.5.2 Synchronisation of Logic and Circuit Simulators

In order to obtain the maximum accuracy from a mixed-signal simulation, two, or more, simulators must generate and respond to signal changes at the correct times [82]. Logic simulators use integral timing units, while circuit simulators use floating point numbers to represent time. Since in practice logic simulation might be two orders of magnitude faster than circuit simulation, the time of a change in a signal passing from a circuit simulator to a logic simulator will be approximated to the next integral time of the logic simulator. Similarly, a change in a signal passing from a logic simulator to a circuit simulator could, if badly arranged, cause the circuit simulator to backtrack in time, in which case a large number of calculations would be thrown away.

A simple solution to the problem above is to run the two simulators independently [82]. Little or no synchronisation occurs, as the output of one simulator becomes the input of the second one. This method is only effective for simple circuits.

Another approach is to take one simulator and to extend it so that to include the functionality of the other. The core simulator can either be the logic simulator or the circuit simulator, but in either case the time control mechanism of the core simulator is dominant and the other simulator is treated as a subroutine to simulate a functional block in the main simulation [82].

The unified approach treats both simulators as equal and ensures that events are passed correctly through a dedicated synchronisation algorithm [82].

The lockstep algorithm couples the two simulators so that both simulators use the same time step. The speed of the simulator is therefore determined by the speed

of the slowest simulator [82]. This technique dramatically increases the number of time steps a simulator must take, which is especially costly to the analogue part of the simulation since each time step causes the matrix to be re-evaluated. Few mixed-signal simulators today employ this technique [90].

Optimistic simulation will allow each simulator to run at its own speed. After a simulation of one circuit reaches no further activity then the simulation can be halted and the other simulation might proceed in turn. If there is little interaction between the two parts of the circuit, the simulation of one part of the circuit may generate new activity for the other as that simulator comes to a halt. More realistically, one simulation will generate activity during the simulation period of the other simulator. This would cause the second simulator to throw away its simulation results generated after this new activity and to backtrack to this new time.

In order to alleviate the inefficiency with the lockstep algorithm, the speed problem, and with the optimistic simulation, the backtracking problem, a hybrid simulation algorithm has been proposed [22]. The hybrid algorithm lets each simulation proceed at its own speed, but it also reduces backtracking and claimed to be more efficient than the lockstep algorithm and optimistic simulation. The hybrid algorithm uses a *backplane*. The *backplane* is a synchronisation and signal parsing mechanism to couple simulators together [22]. The algorithm is described in [82] as follows

1. The next activity time is found for each simulator. In the case of a logic simulator, the next activity time is the time of the next event. For a circuit simulator, the next activity time cannot be predicted accurately: the next time step is an estimate, based on the LTE (Local Truncation Error; the difference between the exact value and the computed value at a time step),

which might subsequently be cut. Therefore, the circuit simulator uses its last solved time as the next activity time.

2. The backplane calculates the start time as the minimum of the next activity times from 1. The simulator with the minimum next activity time is the next simulator to run. The next activity time for the other simulator is the target time. If both simulators return the same minimal next activity time, the start time and the target time are the same. One of the simulators has to be chosen as the next simulator. The simulator that has been idle for the longest time is chosen. If there is again a tie either simulator can be the next simulator.
3. The next simulator runs to the target time calculated in 2. It can stop before target time if it produces a global event. While running, the simulator sends all events that occur before the target time to the other simulator. Events generated after or at the target time, i.e. at the next activity time (refer to Figure 5-4) are not sent because they can be discarded in the next run of the same simulator.

Steps 1 to 3 are repeated until all activity is exhausted or until the user defined finish time is reached.

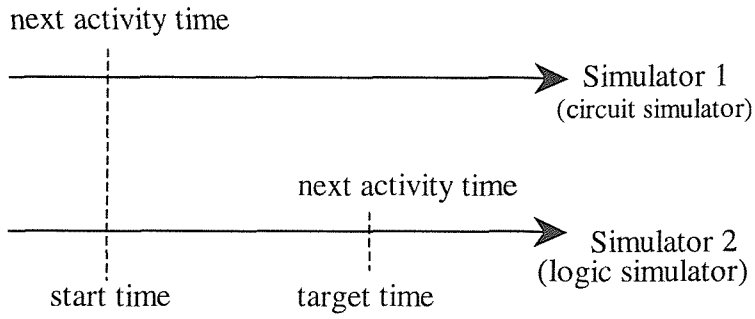


Figure 5-4. Simulator synchronisation times [82].

From the first step of the hybrid algorithm, it can be seen that the circuit simulator can receive global events that occur before or at its next activity time. This is because it is effectively running one step ahead of the logic simulator. Therefore, the next activity time and hence a solved time point of an analogue simulator may have to be discarded. However, backtracking is not a problem if it is limited only to the most recent time point (next activity time), because it is simply a matter of re-evaluating that time point, with a decreased time step.

5.5.3 Mixed Simulator Initialisation

An initial solution is required at the beginning of the simulation, to start the time domain analysis of a mixed-signal circuit [82]. In circuit simulation this initial solution is found by a DC analysis. In logic simulation, the initial state of any undriven node is commonly assumed to be unknown. For digital circuits, initialisation means scheduling all stimulus values at time 0 and evolving the system until all events have expired [90].

For mixed-signal simulation, the existence of unknown states on nodes driving analogue parts of the circuit is not satisfactory. More importantly, an assumption that an unknown state maps always to a specific known analogue value for the purposes of mixed-signal simulation might be inconsistent and might even

cause a DC analysis to fail. Therefore, it is important that a consistent, even though perhaps arbitrary, initial state for the entire mixed-signal circuit is achieved.

5.6 Fault Simulation

Diagnosing the cause of the failures with an integrated circuit that might occur during the design characterisation might be useful before it is in high volume production. If faults are identified and located, the circuit can then be redesigned to alleviate this problem. There are two different approaches proposed in the literature for analogue fault diagnosis [5]: *simulation-before-test* and *simulation-after-test.* Simulation-before-test techniques use a fault dictionary. The faults are then simulated to determine the corresponding responses to predetermined stimuli. Faults are consequently diagnosed by comparing simulated and observed responses. Simulation-after-test techniques, however, begin with failed responses. The failed responses are used to estimate faulty parameter or component values.

Fault diagnosis techniques need fault models. Simulation-before-test techniques are better suited for detecting catastrophic faults and local parametric faults, while they may perform less well in detecting global parametric faults, since for such faults the separation between faulty and fault-free responses is less wide. Simulation-after-test techniques, however, are better suited for detecting problems with global parametric variations and mismatch, and are not well suited for detecting catastrophic faults [5].

In this chapter, mainly analogue fault simulation techniques with regard to simulation-before-test techniques will be investigated.

Analogue fault simulation is the first step to fault coverage analysis, fault grading, fault collapsing, and BIST [28]. Fast fault simulation of analogue circuits is crucially important in terms of speeding up the analogue testing process. There are not many analogue fault simulation algorithms in contrast to digital fault simulation algorithms. This is mainly because there is not yet a standard fault model like the stuck-at fault model of digital circuits available for analogue circuits. If one could establish an analogy between the digital world and the analogue world, it might be possible to come up with some new techniques for analogue fault simulation and analogue testing.

There exist quite a few techniques in order to achieve fast analogue fault simulation. Some of them are: fault dropping/collapsing, in which defects that cause similar changes in the circuit response compared with another faulty circuit response and/or with the fault-free circuit response are considered equivalent; behavioural/macro modelling, whereby parts of the circuit are modelled at a more abstract level, therefore reducing the complexity and the simulation time, and lastly new algorithms such as concurrent analogue fault simulation [23]-[24] and unified approach for fault simulation [25].

Balivada et al proposed a unified approach for fault simulation, where they only considered linear mixed-mode circuits [26]. It was attempted to represent the analogue blocks in the sampled Z-domain using discrete-time state equations in order to provide a common framework for the simulation of both analogue and digital blocks.

Fault simulator used in [25] consists of two modules: The first module performs serial fault simulation of the analogue block, and the other performs parallel or concurrent fault simulation of the digital block. The fault detection is

done at the digital outputs. The drawback with this technique is that analogue fault simulation is done in a serial manner, which is expensive in terms of CPU time.

In this chapter, the discussion is limited to new techniques such as fault dropping to speed up analogue fault simulation. Behavioural/macro modelling issues are discussed in the next chapter.

5.7 CAFS Hierarchical Fault Simulator

CAFS (Concurrent Analogue Fault Simulator) is a hierarchical SPICE-like simulator, originally written in Pascal and later transformed into the C programming language, which has been under development at University of Southampton since the mid 80's.

5.7.1 Concurrent Fault Simulation

Concurrent fault simulation is a fault simulation method where all the faulty circuits along with the fault-free one are simulated at the same time. Redundancies between each faulty circuit and the fault-free one or between each faulty circuit and another faulty circuit can be exploited in order to speed-up the total fault simulation time. For digital circuits concurrent fault simulation is well-documented [1]. More recently researchers have proposed on the application of the concurrent fault simulation method to analogue circuits [23].

In this thesis, concurrent analogue fault simulation in conjunction with new speed-up techniques is investigated for transistor level DC and transient analyses. For DC analysis, all the faulty circuits along with the fault-free one are simulated at the same time while similarities between each faulty circuit and the fault-free one are

evaluated through the use of distance measures. From this data, the faulty responses that are close to and/or far from the fault-free one within a specified threshold are considered redundant. The close faults are those that have very similar responses to the fault-free circuit response hence they are not detectable. The faults that have significantly different responses compared with the fault-free circuit response are said to be detectable faults. Therefore, detectable faults are dropped from further consideration after the DC fault simulation.

In concurrent transient analogue fault simulation, all the faulty circuits along with the fault-free circuit are simulated simultaneously at a time point before simulation proceeds to the next time step. Different time steps may be used to simulate each faulty version of the circuit, in which case the simulation will speed up since some faulty versions might take fewer time steps to simulate. This is distinct from the work done in [24] and [92] as the same time steps were used in these works for the fault-free circuit and other faulty circuits during fault simulation. Further, if the terminal value of the faulty device is close enough to the terminal value of the fault-free device within a specified threshold then the fault-free device values can be reused for the faulty one, thus reducing the CPU time.

5.7.2 Fault Simulation with CAFS

As mentioned before CAFS is a hierarchical SPICE-like concurrent analogue fault simulator. Fault simulation with CAFS can be done at various levels. With "**.options fsim** = n" command (where n takes values from 0 to 3) it is possible to do fault simulation for three different cases. When **fsim** = 0 it means do not carry out fault simulation. When **fsim** equals 1, fault simulation can be done without taking *latency* into account. By latency it is meant that a device is latent when it does not

change its value during N-R iterations or from one time step to another. Similarly when **fsim** takes value of 2, CAFS would consider latent devices and further when **fsim** equals 3 then both latent devices and latent sub-circuits are taken into account, hence the speed-up in the fault simulation in terms of CPU time. For the rest of this chapter, **fsim** option is only set to 1 during simulations, as the main goal is to come up with new speed-up techniques for analogue fault simulation.

5.7.3 Improvements to CAFS

Prior to this thesis study, there was only one way to insert a structural short fault with CAFS into the fault-free netlist describing the circuit to create the faulty version of the circuit for analogue fault simulation; **".short n1 n2 rvalue"** command, where **rvalue** is the resisitive value of the fault between nodes **n1** and **n2**. As circuits get larger it becomes very tedious to use **".short"** command to insert all the possible structural faults into the fault-free circuit netlist manually to create all the faulty versions of the circuit for fault simulation. Another option, therefore, **".options auto"**, is added to CAFS to overcome this problem. When **".options auto"** command is used, CAFS will insert all the possible n(n-1)/2 structural (short) faults automatically into the simulation process, where n is the number of nodes within the circuit. One can argue on the accuracy of these faults, as some of these structural faults might practically not be possible. Depending on the way of inserting faults, CAFS takes the circuit netlist and generates faulty versions of the circuit by inserting one fault at a time into the fault-free netlist to generate a faulty version of the circuit. Automatic structural short faults list generation routine written in C for CAFS is given in appendices, in section 8.1. The technique can easily be extended to cover other faults such as open and parametric faults.

Concurrent fault simulation is well understood for digital circuits [1], [82]. Recently some researchers have tried to apply the concurrent fault simulation algorithm of the digital world to the analogue world [23], [24]. As stated above, no matter whichever analogue fault simulation method is used, fault collapsing and/or dropping is one way to speed up the analogue fault simulation process. To achieve this goal, one needs to compare a faulty circuit response with the fault-free one and/or with other faulty versions of the circuit responses, hence the need for a fast and accurate closeness measurement.

5.7.3.1 Closeness Measurement

In this project discussion of closeness (i.e. distance) measurement is limited to the cases of transient and DC analyses of non-linear analogue circuits only. It is necessary to know how small and/or far is the difference between a faulty response and the fault-free one and/or other faulty responses in order to decide whether to drop and/or collapse this fault (from the fault list). It is also very important to decide when to look at this distance measure and for how many time points in the case of transient analysis and what to compare this distance with to be able to justify the fault dropping/collapsing procedure.

One can carry out the closeness check by either looking at one node in the circuit or all the nodes in which case one needs a measure between two vectors of the same dimension. There are two approaches commonly used for the latter case [82]: taking the difference between the two vectors (which is mathematically correct) or taking the difference for each circuit variable, one at a time. The first approach has been implemented within CAFS.

There are a number of closeness measurement techniques available in the literature but they are not easily applicable to the problem of analogue fault

simulation [93]. In the analogue domain, most widely used closeness measurement has been the Euclidean distance [92], [94]. In [92], the authors used the absolute distance in the multi point sense as a closeness measurement. In [94], the authors used the normalised absolute distance. They applied this closeness measurement for single point closeness check.

5.7.3.2 Single Point Closeness Measurement

Tian et al adopted single point closeness measurement in order to detect the similarity between faulty responses and the fault-free response in DC fault analysis [94]. Let two vectors of real numbers be a vector containing the node voltages and/or branch currents for a faulty circuit and the other vector of the same elements for the fault-free circuit. Let x_f be the vector containing the node voltages and/or branch currents of faulty circuit and x_g be the vector of the same elements for the fault-free (good) circuit. The normalised absolute distance between x_f and x_g is given as [94]

$$d_{fg} = \frac{1}{n}\sum_{k=1}^{n}\left|\frac{(x_f)_k - (x_g)_k}{(x_f)_k}\right| \qquad (5\text{-}2)$$

where d_{fg} is the distance between the vectors x_f and x_g, n is the dimension of the vectors, $(x_f)_k$ and $(x_g)_k$ denote the *k-th* element of the vector x_f and x_g respectively.

In [94], the authors use the distance measure given in (5-2) where they measure the distance between each faulty response and the fault-free response at the end of each N-R iteration. They then order the faults such that the first next fault to be simulated has the minimum distance to the fault-free circuit response. This ordering process is repeated for all the faults. There is no mention in [94] of when to stop this ordering process and how to decide which fault is detectable.

5.7.3.3 Multi Point Closeness Measurement

In [94], the distance check between a faulty response and the fault-free response is carried out after each N-R iteration. The distance might vary during N-R iterations. Yang et al reported that taking single point closeness measurement into account might not accurately detect the distance between faulty circuit responses and the fault-free circuit response [92]. Therefore, in [92], the authors proposed multi point closeness measurement for the DC fault simulation of analogue circuits. The distance between the vectors x_f and x_g is given as follows:

$$d'_{fg} = \frac{1}{M}\sum_{m=1}^{M}\left\|x_g^m - x_f^m\right\| \qquad (5\text{-}3)$$

where d'_{fg} is the distance between the vectors x_f and x_g, $x_g{}^m$ and $x_f{}^m$ are the responses for good circuit and f^{th} faulty circuit at m^{th} N-R iteration and M is the number of consecutive N-R iterations during the DC analysis, over which the closeness check is carried out.

The closeness measurement given in (5-3) was used in [92] in order to decide whether the faulty circuit response is close enough to the fault-free circuit response so that to drop this fault from the fault list at the following iteration step. In [92], however, there is no mention of what threshold they compare this distance with to do the fault dropping.

Most of the time the number of N-R iterations required finding the DC initial conditions is not very large. Therefore, in this project it is chosen to carry out a closeness measurement once the DC initial conditions for the circuit are found. Those faults that have very different responses to the fault-free response can be dropped from further consideration. Note that it is not desirable to drop those faults

that have very similar responses to the fault-free response after DC anaysis as they are not detectable at DC fault simulation step, whereas they might be detectable at transient fault simulation step.

After DC fault dropping set of faults that need to be further simulated in time domain is left. Then, a similar distance check measure is used to attempt dropping faults at transient fault simulation step. This time, close faults also are considered while carrying out fault dropping process.

Assuming that during transient fault simulation x_f is the vector containing the node voltages and/or branch currents of the faulty circuit and x_g is the vector of the same elements for the fault-free circuit, the distance between the vectors x_f and x_g can now be given as:

$$d_{fg}^{TR} = \frac{1}{n}\sum_{i=1}^{n}\left\|x_g^i - x_f^i\right\| \qquad (5\text{-}4)$$

where d_{fg}^{TR} is the distance between the vectors x_f and x_g. x_g^i and x_f^i are the responses for the good circuit and the f^{th} faulty circuit at the i^{th} time point and n is the number of consecutive time points for the closeness measurement.

As x_f and x_g are vectors that contain node voltages and/or branch currents for different nodes in a faulty circuit and good circuit, taking the relative Euclidean norm into account might be misleading for a distance measurement [82]. This is because node voltages might have very different magnitudes and therefore the distance might be dominated by one voltage value. Therefore, it is suggested that the absolute Euclidean norm will give more accurate distance measures [82]. As a part of this PhD study, however, both the relative and normalized absolute Euclidean norms were integrated within the CAFS where the user is able to choose either of these norms for fault dropping. With CAFS, distance check can be done in the

following four ways; single-point single-node, single-point multi-node, multi-point single-node, multi-point multi-node. Single-point means that the closeness check is carried out over one N-R iteration for the DC fault dropping and over one time point for the transient fault dropping, where single-node means that this closeness check is only done for one node within the CUT, which is specified by the user (output node of the CUT for instance). Clearly, the distance check used with multi-point multi-node option will use a number of N-R iterations for the DC fault dropping case and a number of time points for the transient fault dropping case where it will use all the nodes within the CUT.

One need to define a threshold, in order to utilise the closeness measurement results to drop faults. In the next section, therefore, a way to determine this threshold for different distance norms is proposed.

5.7.3.4 Threshold Calculation

Let a_f be the node voltage and/or branch current of a faulty circuit and a_g be the same value for the fault-free circuit in the case of single node closeness check. If it is assumed that the closeness measure is the relative Euclidean norm, and $a_f = ka_g$, then the threshold is given as:

$$th = \|a_g - a_f\| = \|a_g - ka_g\| = (1-k)a_g \qquad (5\text{-}5)$$

where th is the threshold between the faulty value and the fault-free value, and k is an arbitrary constant.

If the closeness measure is the normalised absolute Euclidean norm then the threshold is:

$$th = \left| \frac{a_g - a_f}{a_g} \right| = \left| \frac{a_g - ka_g}{a_g} \right| = |1 - k|. \qquad (5\text{-}6)$$

In the case of multi-node closeness check, each element of the faulty vector might not be perturbed by the same amount (k) as a result of the fault being injected in the fault-free circuit. There will be a vector k consisting of elements (k_1, k_2, ..., k_l) where l is number of nodes in the circuit being simulated. In order to simplify the calculations, however, it is assumed that $k_1=k_2=...=k_l=k$.

In this thesis it is assumed that one wants to drop a fault when the faulty response is within the 5% of the fault-free response or 35% far from the fault-free response. This selection is arbitrary due to lack of agreed standard definitions for analogue circuits.

In order to carry out closeness check between each faulty circuit response and the fault-free circuit response in the DC and transient analyses with CAFS a number of algorithms were developed in C (see appendices, Section 8.1) and implemented in CAFS. Next section describes how these closeness measures are utilised within concurrent fault simulation algorithm implemented in CAFS.

5.7.3.5 Concurrent Fault Simulation Algorithm

In this thesis, structural bridging/short circuit faults between each pair of nodes within an analogue/mixed-signal circuit have been used, which are constructed using the fault-free circuit netlist. If the fault free circuit consists of n nodes then one can say that one has $n(n\text{-}1)/2$ faulty versions of the circuit to analyse as it is assumed that there is a short circuit fault between each pair of nodes in the fault free circuit. To simplify the matters, a resistance value of 10 ohms is assigned to each short

circuit fault. This value can of course be changed if desired. Similarly the work can easily be extended to cover open circuit faults and parametric faults as well.

The steps for the algorithm that has been implemented in CAFS simulator are as follows:

Step 1. Constitute the original fault list, $\beta^0 = \{F^1, F^2, \ldots, F^N\}$, by inserting all possible short faults ($N=n(n-1)/2$) into the fault-free circuit to obtain N faulty versions of the circuit. Here β^0 represents the original fault list and F^k is the kth fault in the fault list. Let x_0^{DC} be the response vector of the fault-free circuit and x_f^{DC} be the response vector of the fth faulty circuit at the at the end of the concurrent DC fault simulation, x_0^i, $i \in [0, \infty)$ be the response vector of the fault-free circuit at the ith time point and x_f^i, $i \in [0, \infty)$ be the response vector of the fth faulty circuit at the ith time point during concurrent transient fault simulation.

Step 2. DC simulate all faulty circuits along with the fault-free one. Now x_0^{DC} and x_f^{DC} are available.

Step 3. Carry out a closeness measurement between each faulty circuit response, x_f^{DC}, and the fault-free circuit response x_0^{DC}. If a faulty response is far enough outside user-defined thresholds then drop this fault from the original fault list. Now the faults to be taken into account for the concurrent transient fault simulation are left.

Step 4. Transient simulate all remaining faulty circuits along with the fault-free one for a number of time points. Now, x_0^i and x_f^i are available.

Step 5. Carry out a closeness measurement between each faulty circuit response, x_f^i, and the fault-free circuit response x_0^i. If a faulty circuit

response is either within or far enough outside user-defined thresholds then drop this fault from the fault list.

Dropped faults are not simulated further during the transient fault simulation. In other words it is implicitly assumed that for a given test stimulus, faults that cause either very similar or sufficiently different behaviour from the fault-free behaviour do not require further analysis. This is because faults that cause very similar behaviour to the fault-free behaviour are not detectable, and the faults that are so far from the fault-free behaviour are said to be detectable. The user may choose the time interval over which the closeness check is to be performed during the transient fault simulation. Clearly, it makes sense to perform the closeness check only once the steady state has been reached (for a periodic stimulus). Similarly, performing a closeness check and then applying a significantly different stimulus would not be sensible as the dropped faults would not be tested against that new stimulus. The faults that cause the simulator not to converge are automatically dropped from the fault list. The reason why for some faults CAFS do not converge during the initial DC analysis is that as CAFS currently utilises only N-R algorithm for solving the equations, N-R algorithm implemented in CAFS is not capable of finding DC convergence for those faults. Therefore, fault coverage figures given in the next section are based on the faults that are convergent rather than the total number of possible structural short circuit faults within the CUT.

5.7.4 Examples

An inverter circuit, Figure 5-5, and the 2-stage Miller opamp, Figure 5-6, the state-variable active filter, Figure 5-8, and the leapfrog filter, Figure 5-9, from the IEEE Mixed-Signal Benchmark Circuits suite [73] is used, in order to validate the

techniques mentioned in the previous sections of this chapter to speed-up the analogue fault simulation.

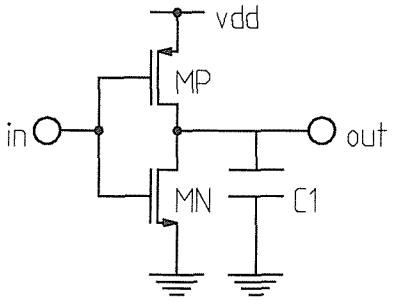


Figure 5-5. CMOS Inverter.

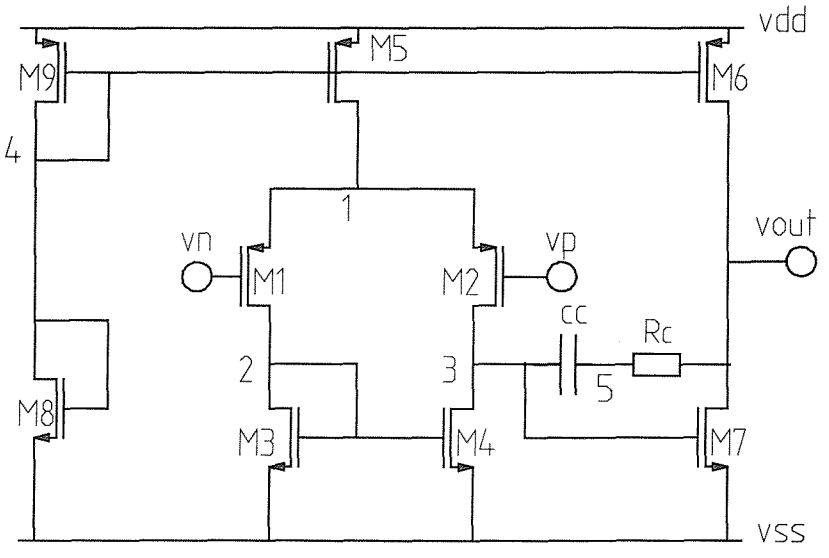


Figure 5-6. 2-stage CMOS Miller opamp [73].

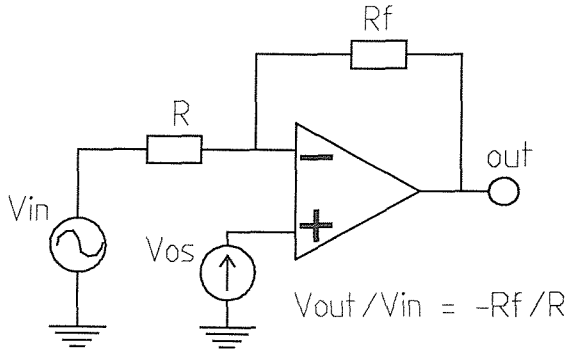


Figure 5-7. Inverting amplifier using opamp of Figure 5-6.

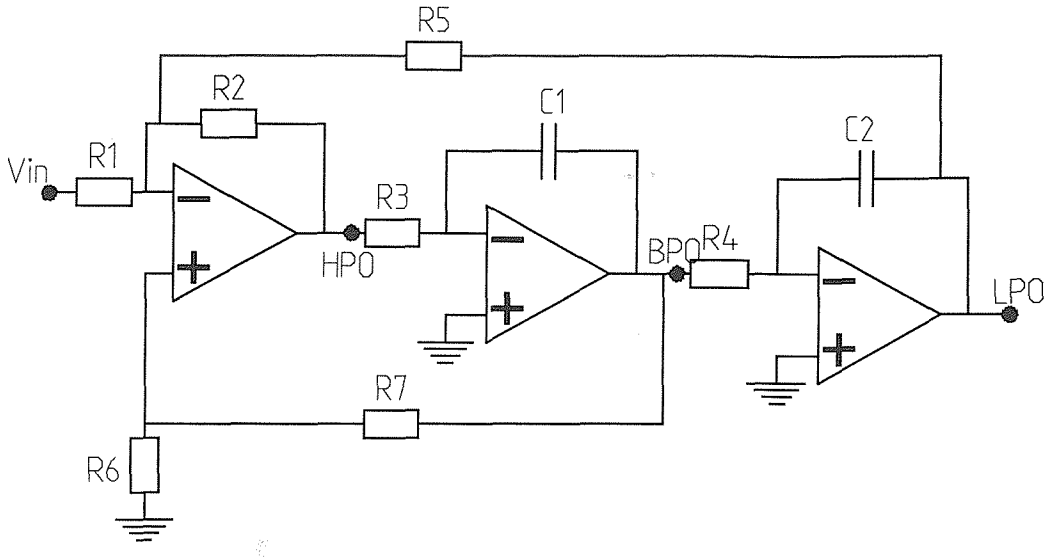


Figure 5-8. State-variable active filter [73].

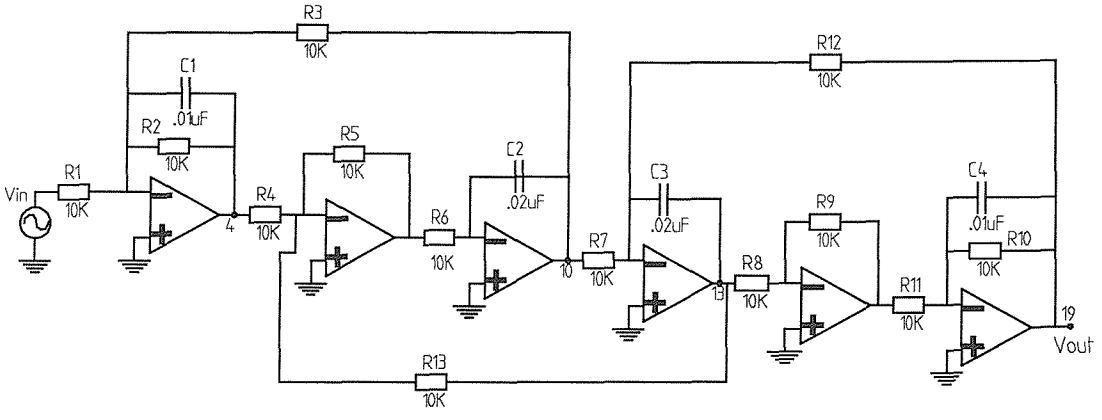


Figure 5-9. Leap-Frog Filter [73].

To demonstrate the validity of the techniques proposed in the previous sections of this chapter let us consider the inverting amplifier given in Figure 5-7, which is constructed using the simple 2-stage operational amplifier depicted in Figure 5-6. A sine wave as the input stimulus for the inverting amplifier is used. All possible distance norms that have been implemented in CAFS are used (see section 5.7.3.3), where looking at the output node for the single point distance checks. During transient concurrent fault simulation, a distance check was carried out on the

second period of the output signal where simulations were run over 10 periods of the input stimulus. The reason why the distance check was carried out over the second period of the output signal is; it was noticed that after the end of the first period, the steady-state was reached for the example circuits were used. This might not be the case for some other circuits where the steady-state is not reached after one period. Therefore, one should start taking distance measurement once the steady state is reached. 55 possible short faults existed for the inverting amplifier circuit, but 7 of them failed to DC converge. Therefore, 48 short faults were left in total to simulate. As mentioned before, 5% and 35% margins for closeness and "farness" are used in order to drop a fault. The speed-up (in terms of the CPU time and the number of device evaluations) and fault coverage for different norms over simulated faults are presented in Table 5-I. Note that the speed-up figures quoted are found simply by dividing the simulation time (in terms of the CPU time and in terms of the number of device evaluations) for concurrent analogue fault simulation with no fault dropping to the same values for concurrent analogue fault simulation with fault dropping cases.

Table 5-I. Fault simulation results for inverting opamp.

opamp	Simulation Time		Speed-up		Fault Coverage				
FD method	CPU (s)	Tot # of Dev. Ev.	CPU	Dev. Ev.	Total # of sim. faults	# of far faults (DC)	# of close faults (TR)	# of far faults (TR)	TOTAL FC (%) (DC+TR)
seucl snode	13.2	39753	4.6	6.6	48	28	8	12	100
seucl mnode	156	680067	0.4	0.39	48	28	11	0	81.25
meucl snode	13.2	39753	4.6	6.6	48	28	8	12	100
meucl mnode	156	680067	0.4	0.39	48	28	11	0	81.25
snabsl snode	42	171684	1.4	1.5	48	18	14	3	72.9
snabsl mnode	48.2	200961	1.3	1.3	48	18	11	0	60.4
mnabsl snode	23.7	90009	2.6	2.9	48	18	11	9	79.16
mnabsl mnode	27.6	105984	2.2	2.5	48	18	2	14	70.8
NO FD	60.7	263151							

Abbreviations given in Table 5-I, which are used for different closeness measures, are clarified in Table 5-II

Table 5-II. Explanation of abbreviations used in Table 5-I.

FD method:	Fault Dropping method
seucl snode:	Single-point single-node relative Euclidean distance
seucl mnode:	Single-point multi-node relative Euclidean distance
meucl snode:	Multi-point single-node relative Euclidean distance
meucl mnode:	Multi-point multi-node relative Euclidean distance
snabsl snode:	Single-point single-node normalised absolute Euclidean distance
snabsl mnode:	Single-point multi-node normalised absolute Euclidean distance
mnabsl snode:	Multi-point single-node normalised absolute Euclidean distance
mnabsl mnode:	Multi-point multi-node normalised absolute Euclidean distance
NO FD:	No Fault Dropping

As can be seen from Table 5-I, for "seucl mnode" and "meucl mnode" norms the CPU time spent with no fault dropping seems to be larger than the CPU time spent with fault dropping option on. This is due to the way the next time step is

calculated within CAFS for each circuit during the concurrent fault simulation. For "seucl mnode" and "meucl mnode" norms, when fault dropping option is on CAFS seems to spend more time on faults such as vout-ground short and vp-vout short (for the inverting opamp given in Figure 5-7) while different initial time steps are employed compared to the case of concurrent fault simulation with no fault dropping option. The time step selected for those faults make CAFS to backtrack in time while cutting the time step each time in order to find a convergence. When the time step is cut too much (near zero) CAFS then drops this fault from further simulation. This, in effect, increases the CPU time spent for simulation. On the contrary, even though when no fault dropping technique is used, some faults are dropped earlier due to the time step selection within CAFS.

In order to demonstrate the speed-up for different CUTs, for the above-mentioned benchmark circuits single-point single-node Euclidean norm as the distance norm, 5% and 35% margins for closeness and "farness" were used. The transient analysis was run over 10 periods of the input stimulus and the distance check was carried on the second period of the output signal again, where it was observed that after the first period of the input stimulus the steady state was reached. Note that the choice of which norm and what threshold to so that to drop a fault from further consideration is totally user-defined.

Table 5-III represents the speed-up and the fault coverage results for the different benchmark circuits used. As can be seen from Table 5-III, depending on the choice of the distance norm used (in this case we use "seucl snode" distance norm) up to 100% fault coverage, and up to 6.6 speed-up (in terms of the number of device evaluations) over fault simulations with no fault dropping is possible. Note that fault

collapsing, which might further increase the speed-up in the analogue fault simulation, is not considered in this PhD study.

Table 5-III. Speed-up and Fault coverage for benchmark circuits.

	Speed-up		Fault Coverage				
Benchmark	CPU	Dev. Ev.	Total # of simulated faults	# of far faults (DC)	# of close faults (TR)	# of far faults (TR)	TOTAL FC (%) (DC+TR)
INV	3.1	3	6	2	3	1	100
OPAMP	4.6	6.6	48	28	8	12	100
FILTER	4.2	4.6	95	34	20	35	93.7
LEAPFROG	4.7	4.65	378	97	129	133	94.9

5.7.5 Fault Simulation using Built-in Current Sensor (BICS)

In this section some fault simulations will be carried out using the process variation independent BICS circuit developed in Chapter 3 of this thesis. The state-variable filter given in Figure 5-8 will be used as the CUT.

It has been observed from the previous section that some of the faults within the state-variable filter circuit are not detectable while most of them are using CAFS. The faults that are not detected for the filter circuit include "bpo-4" short and "2-ground" short (see Figure 5-8). If one carefully observes the circuit given in Figure 5-8, the first fault is a short across resistor R4, while the second fault is a short across resistor R6 in the circuit. It is obvious from the circuit configuration that these faults do not contribute very much on the output response and the current drawn from the supply. Fault simulations using HPSICE and BICS were carried out for these faults

while trying to monitor the dynamic supply current (note that TM process parameter set was used during simulations). It has been found due to the reason given above the BICS correctly monitored the dynamic supply current for both faults. The results for these faults are exactly the same as that given in Figure 3-23 (see page 71), as the current drawn from the supply did not change from the fault-free case for these two faults.

Faults simulations then carried out for some of the structural short faults that are detectable using CAFS for the filter circuit. The first fault is "lpo-1" short where resistor R5 in Figure 5-8 is shorted. The simulation result for this fault using HSPICE and BICS circuit is given in Figure 5-10. As can be seen from the figure the dynamic CUT current for this fault case is nearly zero, which is accurately monitored by BICS circuit.

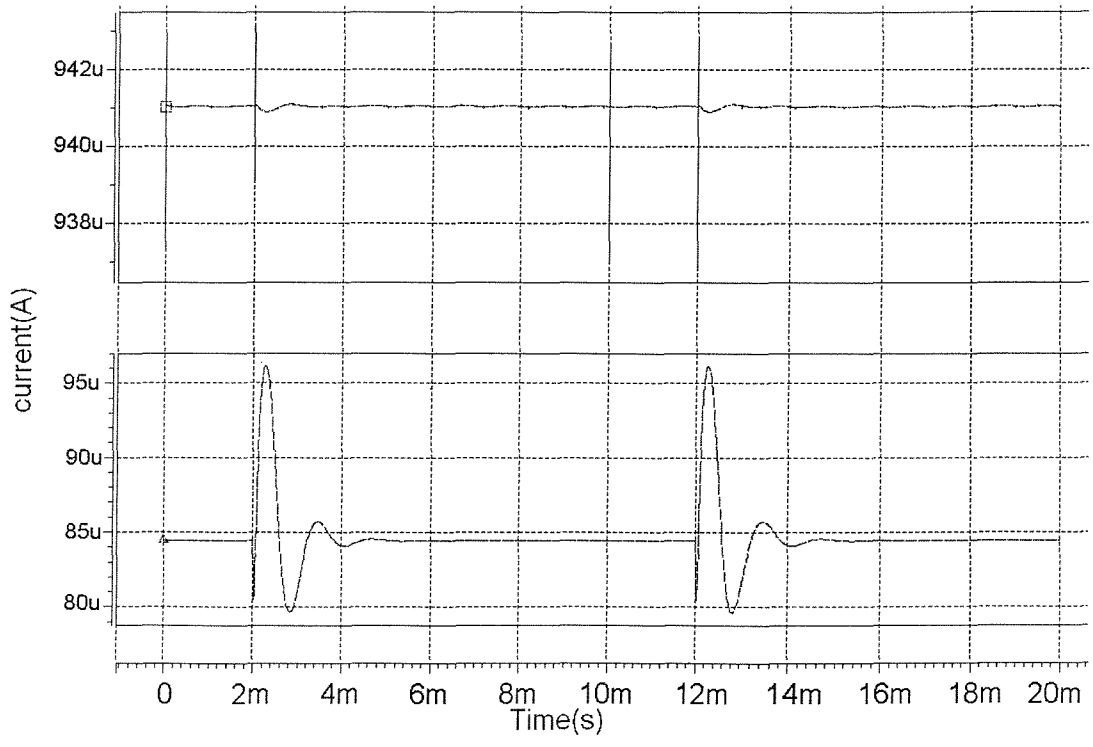


Figure 5-10. The faulty current drawn by the CUT (top) and the monitored current with the BICS for TM parameter set for "lpo-1" short fault.

The second fault for the filter is chosen to be "vin-bpo" short fault. The fault simulation result using HSPICE and BICS is given in Figure 5-11. In this fault case the faulty dynamic CUT current is larger than the fault-free one by around 30% (see Figure 3-23 for the fault free dynamic CUT current). Note that the faulty dynamic current is also somewhat non-linear. As can be seen from Figure 5-11 BICS monitors this faulty dynamic current accurately as well.

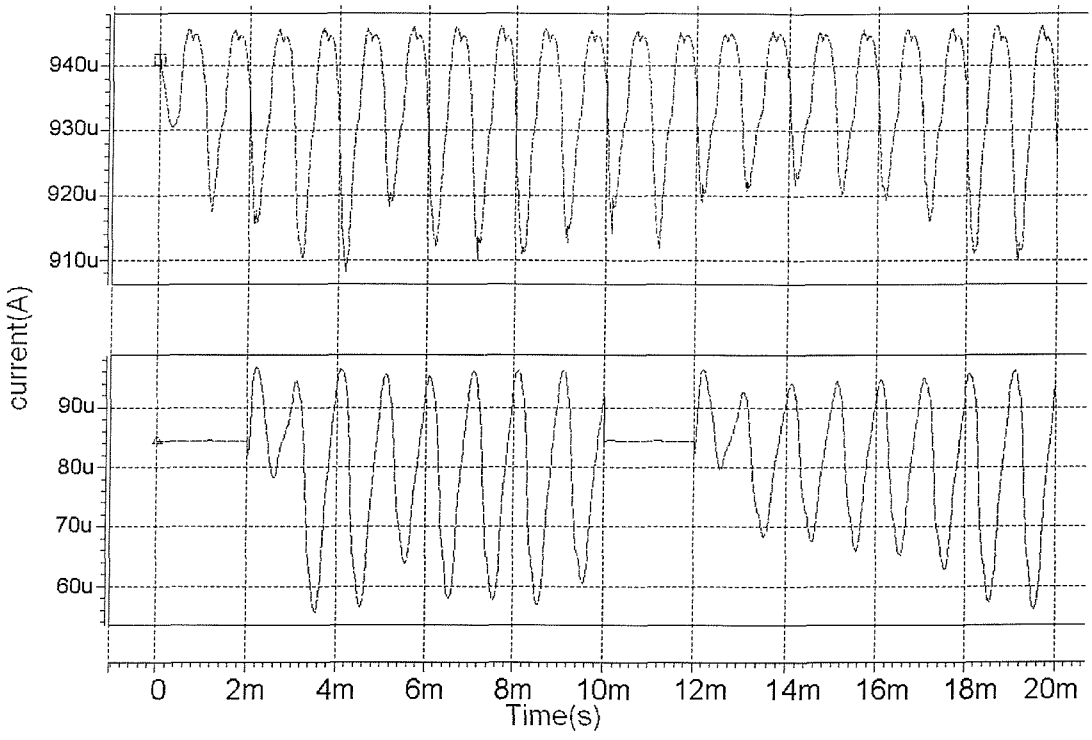


Figure 5-11. The faulty current drawn by the CUT (top) and the monitored current with the BICS for TM parameter set for "vin-bpo" short fault.

Therefore, it is fair to claim that the BICS circuit designed in Chapter 3 of this thesis can be used in an analogue self-test technique for monitoring the dynamic supply current to the CUT. Note that, as can be seen from the example cases given above, it is obvious that using the BICS or more accurately monitoring the dynamic CUT current will not lead to detect all the possible short circuit faults within an analogue circuit.

6 BEHAVIOURAL/MACRO MODELLING

6.1 Introduction

As transistor sizes keep shrinking, Integrated Circuits (ICs) have been growing in size and complexity. This growth in ICs also causes the testing problem to be much more difficult. For digital circuits the problem of testing can be simplified by using standard fault models and relatively fast fault simulation. Faults in digital circuits can be modelled as stuck-at, bridging and open faults [1]. These structural faults can then be used to generate test vectors. The objective of a test program for digital circuits translates into whether or not a fault exists using the smallest possible number of test vectors [35]. While one test might detect more than one fault, each fault might be covered by more than one test. Therefore, test pattern generation is the process of selecting an optimal set of tests from all possible input patterns. This optimal test pattern selection can be done in an ad-hoc manner for small and simple circuits. For larger circuits the optimal set of tests, which consists of the smallest number of tests that gives the highest fault coverage, can be chosen using algorithms such as D-algorithm or PODEM [1].

A test pattern is evaluated by looking at its fault coverage. All faults detected with this pattern can be dropped from further consideration. Fault simulation is done

for the assessment of the fault coverage. There are a number of fault simulation techniques used for digital circuits. Serial fault simulation is perhaps the simplest method. In serial fault simulation, for each fault a faulty copy of the circuit with the fault inserted into is created. Then, all the faulty copies of the circuits along with the fault-free one are simulated with the given test pattern one at a time. If the output of a faulty circuit differs from the fault-free one, this fault is said to be detectable for that test pattern.

Another fault simulation technique for digital circuits is concurrent fault simulation [1]. The differences between the faulty and the fault-free circuit behaviours might be relatively small. Therefore concurrent fault simulation is targeted to avoid redundant element evaluation when the fault-free and faulty behaviours are the same, hence reducing the computational effort required to simulate all faulty circuits.

Most of the fault simulation techniques developed for digital circuits are not easily applicable to analogue and/or mixed-signal circuits (unless otherwise stated, in the rest of this chapter the term *analogue* will cover both analogue and mixed-signal). This is due to the fact that even though structural faults such as open and short circuits can be identified, they do not affect analogue circuits in a binary manner. Therefore, test pattern generation for these circuits has mainly been done in an ad hoc manner.

The main difficulty while generating test patterns for analogue circuits is perhaps the fault simulation. It has been shown that the simulation of analogue circuits is at least two orders of magnitude slower than that of similarly sized digital circuits using traditional methods [22]. This is due to the fact that digital circuit

simulators use less complex algorithms (as explained in the previous chapter) compared with transistor level simulators.

One way to speed up the fault simulation is to model the faulty circuits at a higher level so as to reduce the complexity of the model and hence the CPU time spent for the simulation. Therefore, research has recently focused on how to model faulty circuits at a more abstract level for efficient fault simulation [35], [95], [107]-[109]. The lack of standard fault models for analogue circuits, however, makes the problem of modelling the faulty analogue circuit block at a higher level more and more difficult.

Using a fault dropping technique in conjunction with concurrent analogue fault simulation was discussed in the previous chapter. This chapter is mainly concerned with behavioural/macro modelling as another way to speed up the analogue fault simulation.

6.2 Previous Work on Macromodels for Analogue Circuits

Extensive fault simulation is needed for fault grading, yield estimation, test vector generation, etc. Fault simulation at the transistor level for analogue circuits is computationally very expensive. Therefore, one way to reduce this high simulation cost is to partition a large analogue circuit into smaller functional blocks such as opamps and replace each functional block with its *macromodel* or describe each block using mathematical equations, which is called a *behavioural model*. This solution is sometimes called *hierarchical fault simulation* in the literature [95]. The

reduction in simulation time may be achieved at the expense of accuracy with the simulation results.

The word *macromodel* usually refers to a compact representation of a circuit that captures those features that are useful for a particular purpose while discarding redundant information [96]. Macromodels developed for SPICE-like simulators are basically electrical networks containing devices such as a voltage-controlled voltage source instead of the full transistor network and with fewer nodes than the original circuit.

Many circuits are designed in a modular style, in which functional units are connected to achieve the design specifications. The behaviour of the whole circuit is determined by how the individual units interact with each other, while what happens inside each is unimportant in terms of the behaviour of the entire circuit. The accuracy of a macromodel must, therefore, be defined in terms of how closely its input-output behaviour matches that of the original unit [96].

Since the early 1970s a number of macromodels have been developed mainly for integrated operational amplifier circuits (opamps) [34], [36], [37], [95]-[103]. Boyle et al presented a macromodel for integrated bipolar opamp circuits [34]. This macromodel was six times less complex (in terms of the node count) than the original opamp circuit, and the macromodel provided simulated circuit responses that had run times of an order of magnitude faster than the device-level model.

The derivation of component values for the Boyle macromodel is not, however, straightforward. Some parameters are modelled using unbalanced input devices and other parameters interact. Therefore, in [97] a new macromodelling approach was proposed. A modular macromodel approach was suggested, in which a macromodel was derived simply from the published data sheets. Individual

parameters were modelled separately and the results were combined to provide the output response. Since the parameters were separate they did not interact and only those required were included.

Macromodels that are developed for analogue circuits have proved useful in terms of simulation time [34], [36], [37], [96]-[103]. Therefore, recent research has focused on how to capture the effect of a fault that might occur within an analogue circuit in the macromodel [35], [95], [107]. In [95], a fault macromodelling approach for analogue circuits was proposed. The fault macromodelling problem was formulated in terms of deriving the macro parameter set, B, based on the performance parameter set, P, (gain, bandwidth, samples on the frequency or time response curves, etc.) of the transistor-level faulty circuit. The accuracy of the macromodel was evaluated by checking the consistency of the performance parameter set, P, between the transistor level circuit and the macromodel.

Two steps are needed to obtain the macromodel for a functional block within an analogue circuit [95]:

1. Perform a transistor level fault simulation for each faulty circuit to obtain the value of the performance parameter set P.
2. Map each performance parameter set P to the corresponding macro parameter set, B. This is referred to as the *parameter mapping*.

It is assumed that the transistor-level fault list is given and the macromodel structure and the performance parameter set, P, to be matched are predetermined by the circuit designer.

There are several ways to do parameter mapping. One simple approach is based on analytical design equations that express the macro parameter set, B, as analytical functions of the performance parameter set, P, and the value of B is derived by function evaluation. As analogue ICs are getting more and more complex,

this approach is becoming more difficult. Another simpler approach is to build an empirical mapping function, $B=F(P)$, based on a large number of data pairs (P, B), which are referred to as the *training set* [95]. Usually the training set is generated by randomly selecting M out of the N performance parameter sets for all the faulty circuits obtained by the transistor-level simulation and then the value of the macro parameter set B for each selected P is derived. The derivation of each data pair usually requires multiple runs of macro level simulation [95].

Macromodelling in general and fault macromodelling in particular, using SPICE-like languages, nevertheless, have been shown to be very difficult [34]-[38], [95]-[104], [109]. Therefore, another easier and perhaps more efficient way of modelling analogue circuits at a higher level is necessary.

6.3 Behavioural Modelling

A behavioural model describes a circuit block in terms of mathematical equations modelling the functionality of the block, for example, in terms of the input-output relationship. Behavioural modelling has been used for speeding up analogue simulation in general [105] and analogue fault simulation in particular [35], [107]-[109]. Analogue circuits were modelled behaviourally in the C programming language in [105]. Broyden's method was used to formulate and solve the model equations in a custom simulator. Broyden's method was originally proposed in [106] as an algorithm for the solution of systems of nonlinear equations. The main drawback with the work described in [105] is that since the technique does not require derivatives it cannot be used for small-signal analysis, which by definition depends on the computation of derivatives [82], [83], [105].

In [107], Chang et al presented a behavioural fault model derived from a macromodel for the CMOS operational amplifier from the IEEE Mixed-Signal Benchmark Suite [73] (Figure 6-1). The fault macromodel was developed via using DC-sweep analysis. The DC behaviour of the benchmark opamp operating in inverting, non-inverting and unity gain amplifier configurations, as shown in Figure 6-2, was first investigated under different faulty conditions. Single transistor catastrophic faults, bridging/short and nearly open faults, and parametric faults with W (channel width), L (channel length) and V_t (threshold voltage) varied by $\pm 10\%$ were used for each transistor. Then an attempt was made to group the different faulty behaviours. By comparing the fault-free offset voltage measured at the inputs of the opamp operating in one of the three configurations with the equivalent faulty circuits, four different equivalent fault types were derived [107]: M4 drain-to-gate short (Type I), M5 drain-to-source short (Type II), M7 drain open (Type III), and M5 drain-to-source short (Type IV). The first three fault types existed for the opamp operating in the inverting configuration; the Type IV fault group was found for the non-inverting configuration.

The input offset voltage (measured between the non-inverting and inverting inputs of the opamp in the closed-loop configurations) and the output voltage versus the input voltage for the fault-free opamp operating in three configurations were determined by carrying out HSPICE simulations and are shown in Figure 6-3, Figure 6-4, and Figure 6-5, respectively.

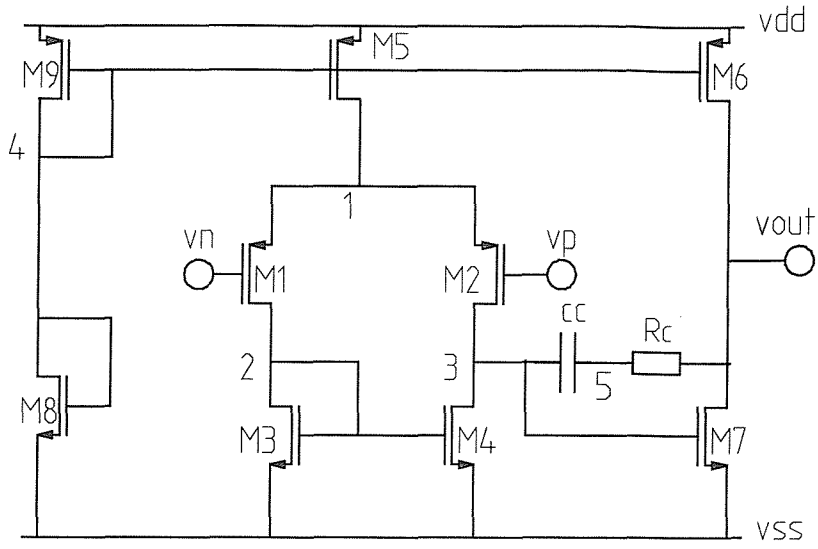


Figure 6-1. The 2-stage CMOS Miller opamp used in [107] for behavioural fault modelling.

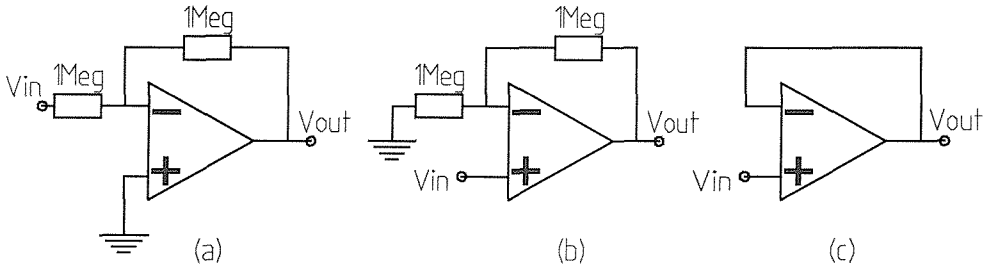


Figure 6-2. Three different configurations used in [107] for the benchmark circuit given in Figure 6-1: (a) Inverting amplifier, (b) non-inverting amplifier, and (c) unity gain buffer.

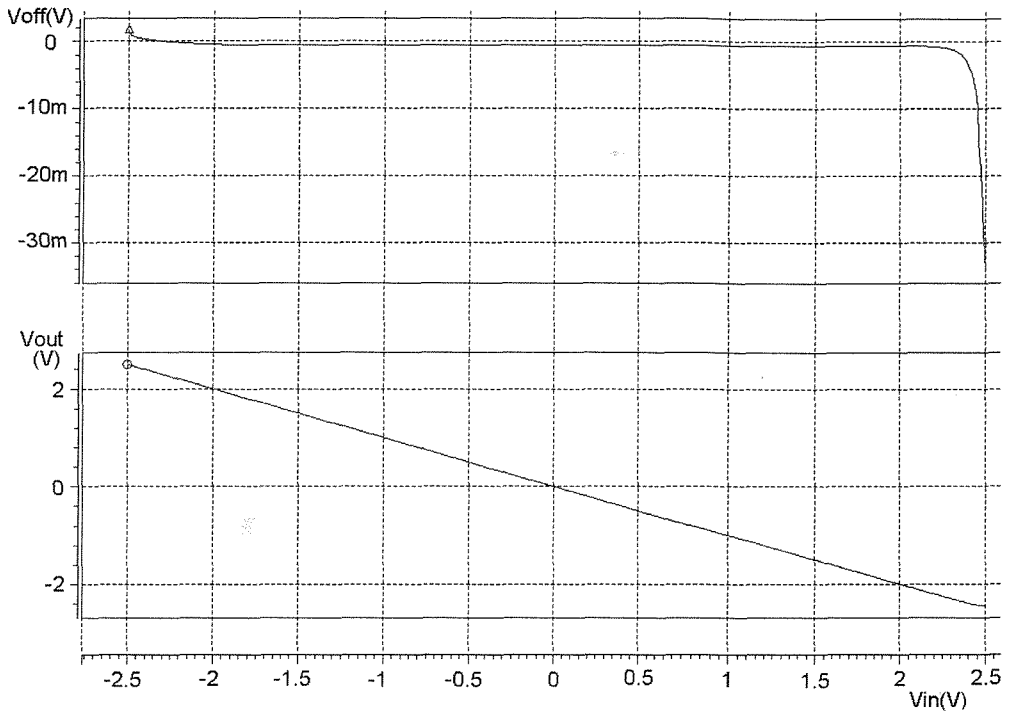


Figure 6-3. Input offset voltage and output voltage versus DC-sweep input voltage for the fault-free inverting amplifier.

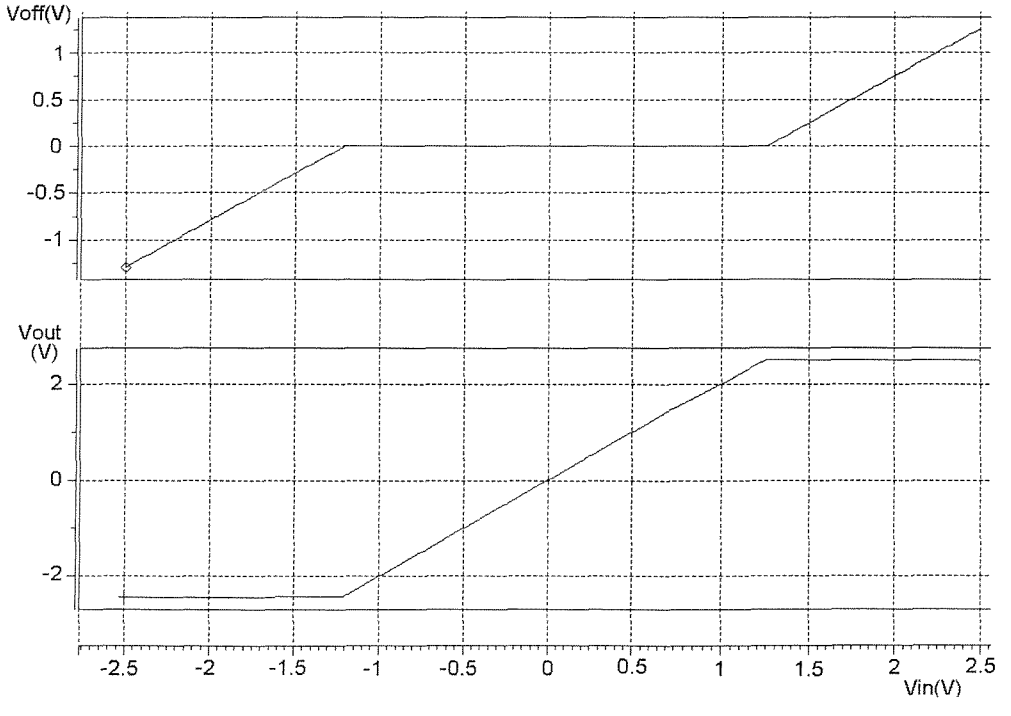


Figure 6-4. Input offset voltage and output voltage versus DC-sweep input voltage for the fault-free non-inverting amplifier.

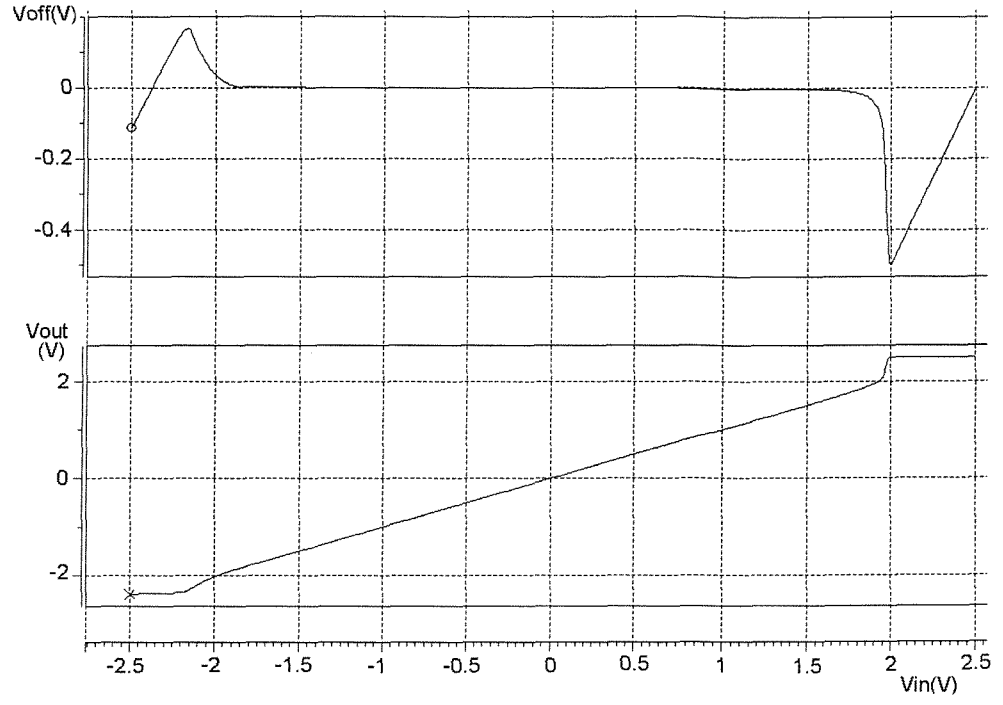


Figure 6-5. Input offset voltage and output voltage versus DC-sweep input voltage for the fault-free unity gain buffer.

Next, the input offset voltage and the output voltage for each fault group with respect to the input voltage were found by HSPICE simulations and are shown in Figure 6-6, Figure 6-7, Figure 6-8, and Figure 6-9, respectively.

As can be seen from Figure 6-7 and Figure 6-9, output responses obtained by Type II fault and Type IV fault are quite similar to the fault-free output responses given in Figure 6-3 and Figure 6-4. Note that, Type II and Type IV input offset voltages are somewhat different from the fault-free responses. The input offset voltage has a small DC level for Type II faults, but has a non-linear characteristic for Type IV faults.

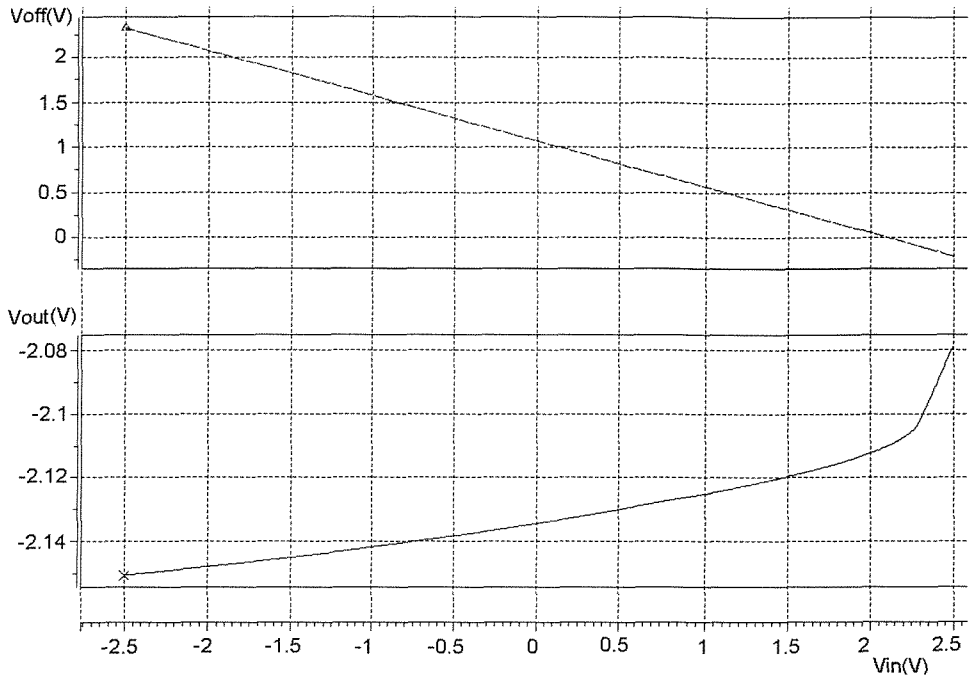


Figure 6-6. Input offset voltage and the output voltage for the Type I fault (M4 drain-to-gate short fault for the inverting amplifier configuration).

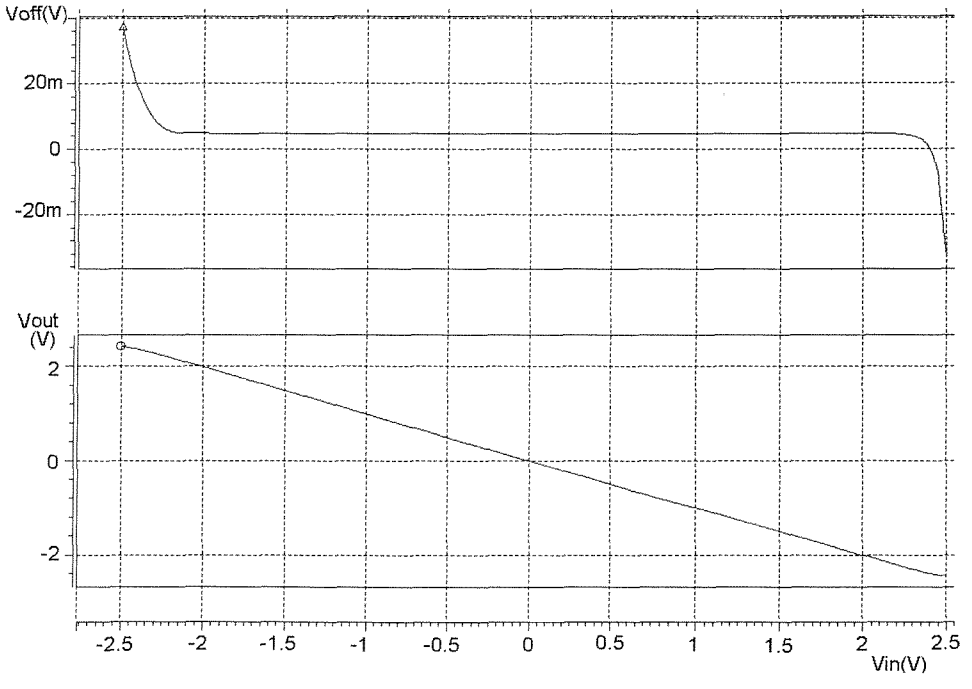


Figure 6-7. Input offset voltage and the output voltage for the Type II fault (M5 drain-to-source short fault for the inverting amplifier configuration).

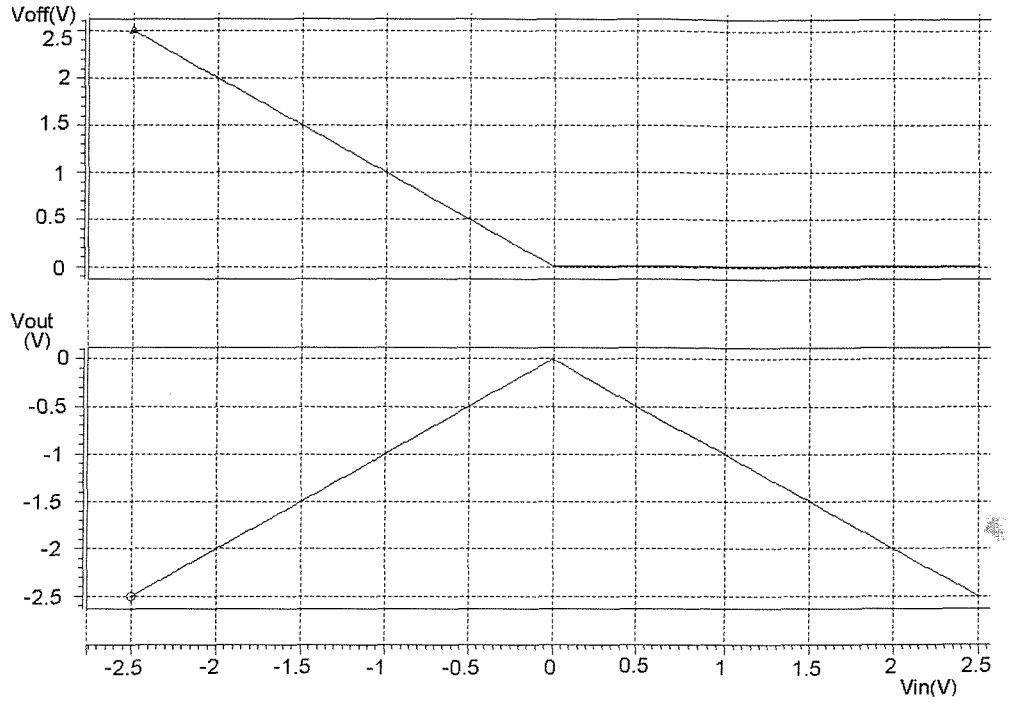


Figure 6-8. Input offset voltage and the output voltage for the Type III fault (M6 open drain fault for the inverting amplifier configuration).

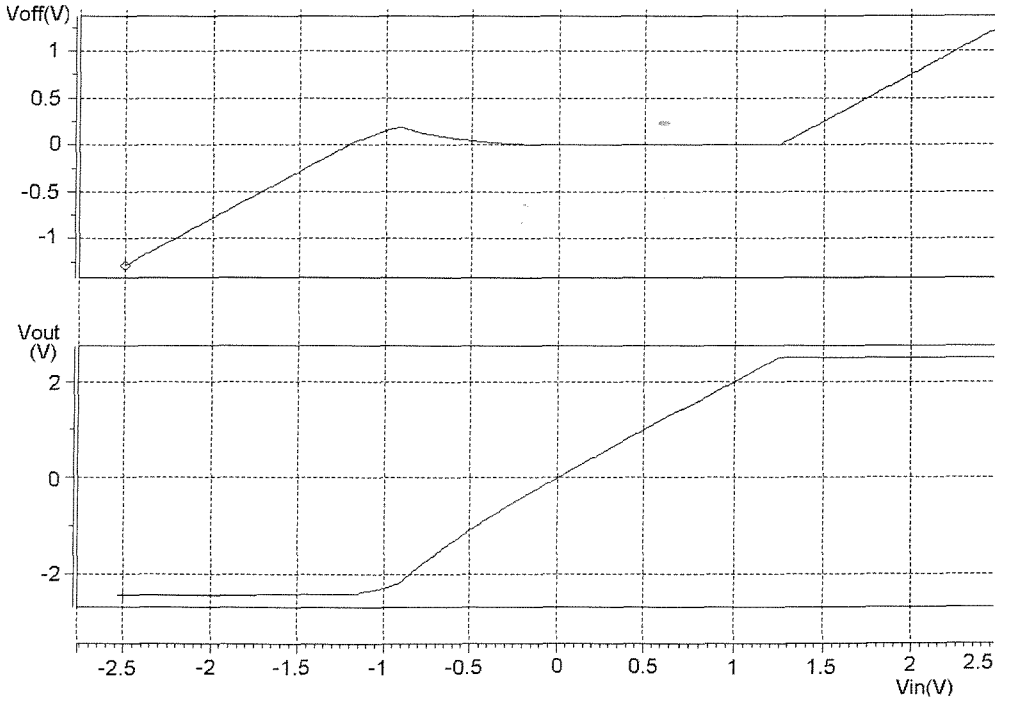


Figure 6-9. Input offset voltage and the output voltage for the Type IV fault (M5 drain-to-source short fault for the non-inverting amplifier configuration).

The remaining two faults have very different characteristics to the fault-free equivalents for both input offset voltages and output voltages. It can be concluded from the figures that a Type I fault causes the inverting amplifier output to be nearly stuck-at a negative voltage near to the negative supply voltage level. A Type III fault causes the inverting amplifier output to have a non-inverting characteristic for the negative values of the DC input signal, and an inverting characteristic for the positive values of the DC input signal. As can be seen from the figures above, the input offset voltage at the inputs of the opamp has a linear characteristic for Type I faults, and a piecewise linear characteristic for Type III faults.

Next, the macromodel given in Figure 6-10 for the inverting opamp was used to derive the input output relationship under fault conditions. This relationship is given in [107] as:

$$V_{out} = A_{CL}[(1+m)V_{in} + k] \qquad (6\text{-}1)$$

where A_{CL} represents the closed-loop gain for the opamp, the parameters m and k are given in [107] as:

$$m = \frac{-R2}{D+R2} \qquad (6\text{-}2)$$

and

$$k = aVos + bVdd + cVss \qquad (6\text{-}3)$$

where

$D = B(R2 // Ro // Rdd // Rss)$,

$B = \left(\frac{A}{Ro} - \frac{1}{R2}\right)(Rid // R1 // R2 // 2Ricm)$,

$$a = \frac{R2 // D}{A_{CL}(R1 // R2 // 2Ricm // BR2)},$$

$$b = \frac{SF}{A_{CL} Rdd},$$

$$c = -\frac{SF}{A_{CL} Rss},$$

$$A_{CL} = -\frac{R2}{R1},$$

$$SF = Rdd // Rss /// Ro // (R2 // R11) // \frac{Ro(R11 // R2)}{AR11},$$

$R11 = R1 // 2Ricm // Rid$, and A represents the open-loop gain.

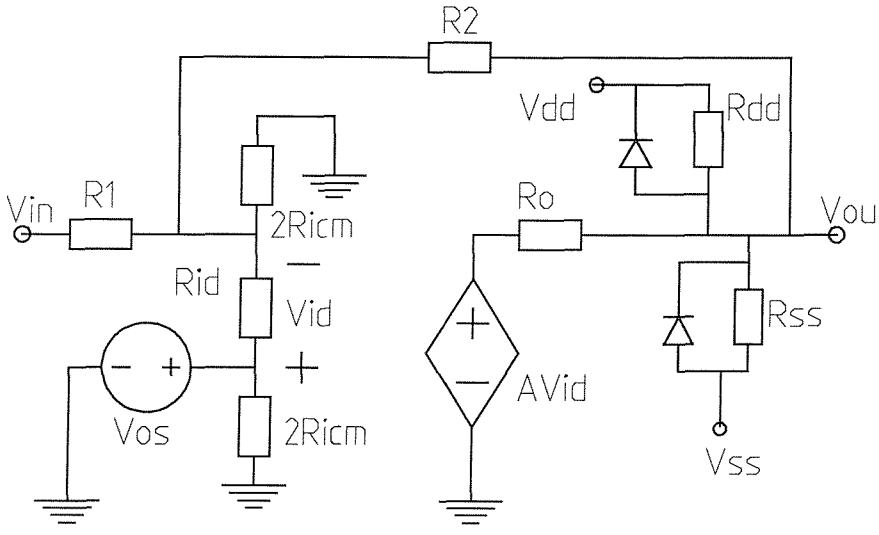


Figure 6-10. Macromodel used in [107] to derive the input-output relationship for the closed loop inverting opamp.

The non-ideal effects such as the input offset voltage, *Vos*, the finite open-loop gain, *A*, and the finite input and output resistances, *Rid* (differential mode input resistance), *Ricm* (common mode input resistance), *Ro* (output resistance), and the resistances from output node to the supply rails (*Rdd* and *Rss*) that model output

stuck-at faults were taken into account while deriving equation (6-1) for the input-output relationship. Note that for the fault-free ideal opamp case *Rid*, *Ricm*, *Rdd*, *Rss*, and A would be infinite, *Vos*, and *Ro* would be zero, hence $m \to 0$, and $k \to 0$. When a fault causes the output to be stuck-at some voltage level, $D \to 0$, therefore $m \to -1$, and k is the value of the stuck-at output voltage, while the closed-loop gain, A_{CL}, is assumed to be unity. As it is dealt with elsewhere [107], the derivation of equations given above will not be discussed in this thesis.

In [107], the current limiting effect was also modelled. This is due to the finite supply voltage at the output of the opamp. It was claimed that the model covered all the parametric faults and 92.5% of the catastrophic faults that were considered. The model cannot model the M4 drain-to-gate short, M5 drain-to-source short, M1 open-gate faults for the non-inverting amplifier and M2 drain-to-gate short, M4 drain-to-gate short, M5 drain-to-source short, M1 open gate, M3 open source and M5 open gate faults for the unity gain buffer. (All the transistors, M1-M5, are those given in Figure 6-1). The accuracy of the model was verified against transistor level simulations. To do that, DC and AC fault simulations using both transistor level model and behavioural model were carried out on the inverting amplifier for the parametric faults only, where the threshold voltage, channel length and channel width were varied for each MOS transistor by ±50%. Good correlation between the simulation results of the two models was obtained for DC and AC analyses (lower frequencies). For the frequencies above the unity gain frequency of the opamp in the AC simulations, average of 10.4% error was reported for the opamp output voltage. There is no mention, however, of how the behavioural model is implemented and simulated in [107].

In [108], behavioural models written in standard VHDL were used to describe analogue blocks in order to speed up the analogue fault simulation. A PLL circuit was partitioned into smaller sub-blocks. An automatic modelling tool was used to generate the most of the behavioural models written in VHDL for these blocks. As VHDL is event driven, it is not very suitable for behavioural models of analogue circuits. In the next section, previous work on behavioural modelling using HDLs with emphasis on analogue circuits, which has been done in the literature so far, will be discussed.

6.4 Behavioural modelling using HDLs

Currently two widely-used standards for modelling digital designs are VHDL [111], and Verilog [112]. For analogue modelling, the standard has been usually (transistor level design) using Spice-like languages. Now, extensions to VHDL and Verilog extend these HDLs to analogue design. Therefore, with these mixed-signal HDLs, it should be possible to design systems at any level from transistors to behavioural descriptions for both analogue and digital.

Although one has been able to use analogue behavioural languages for some time, VHDL-AMS and Verilog-AMS are the first languages that attempt to bring analogue and digital HDLs together. In the past, the languages used for the analogue and digital portions of a design were either completely different or proprietary and focused towards the analogue designer. Given a choice, one probably would not write half a program in C and the other half in Fortran. The analogue and mixed-signal extensions to VHDL and Verilog will help to alleviate the multiple-language problem [110].

Top-down design provides a smooth path from initial specification to physical layout. Certainly, many spots in the path could trip one up. On the digital side, behavioural logic synthesis is just becoming available, but only for Digital Signal Processors (DSPs) and other specialized types of systems, such as Field Programmable Gate Arrays (FPGAs). One usually needs to translate a behavioural HDL design to the Register Transfer Level (RTL) before logic-synthesis tools can implement the design. Timing issues may create problems at the layout stage.

In the analogue world, top-down design has even fewer steps. Presently, only a few specialized tools can synthesize analogue designs from behavioural-level descriptions. These tools tend to exist for filter design and related functions. VHDL-AMS and Verilog-AMS set standards for hardware descriptions that may eventually result in the growth of analogue synthesis tools. However, their utility is aimed at helping out with system specification and simulation. One will need to eventually translate the behavioural hardware description into a functional circuit description suitable for implementation at the board or IC level.

Analogue HDLs support the description of systems of differential and algebraic equations (DAEs). The solution of these systems varies continuously with time. Today's analogue HDLs support both structural composition and conservation semantics, in addition to behavioural descriptions. Examples of such languages are FAS [113], SpectreHDL [114], and Verilog-A [115].

Mixed-signal design has depended on the use of separate HDLs for the analogue and digital parts or, again, on proprietary languages. Mixed-signal languages support both event-driven techniques and differential and algebraic equations. Simulators in this category are MAST/Saber [116], VeriasHDL [116], AdvanceMS [113], Hamster [117].

In this thesis behavioural modelling using analogue HDLs including MAST and VHDL-AMS is taken into account. A brief summary of VHDL-AMS can be found in Chapter 8 (Appendices).

6.5 Behavioural Fault Modelling using Analogue HDLs

Since VHDL-AMS was standardised in 1999 there has been limited work done on fault modelling using VHDL-AMS. One reason for the limited progress is perhaps that there was not yet a robust VHDL-AMS simulator available at the time of writing this thesis that had all the VHDL-AMS constructs, such as procedural statements, implemented in it.

Perkins et al attempted to use an analogue VHDL for fault modelling and simulation with very limited success [35]. The authors used the HDL-A modelling language with the ELDO simulator from Anacad (now a part of Mentor Graphics). In [35], the macromodel developed in [36] (see Figure 6-12) was used for the simple two-stage CMOS opamp shown in Figure 6-11. In Figure 6-11, the numbers in the circles denote the node numbers.

In Figure 6-12, Rin is the differential mode input resistance, iout is the output current, itail is the sum of the tail current through M5 and biasing current through Rbias in Figure 6-11, Rout represents the finite output resistance and Acl is the closed loop gain of the opamp. In total 51 bridging/short faults and open circuit faults were inserted [35]. *Fault grouping* (or sometimes called fault collapsing) is the process of grouping the faults that cause the circuit to behave in a very similar way, and only take one of those faults into account during the fault simulation. Fault

grouping results in reduced number of fault simulations hence faster fault simulation times.

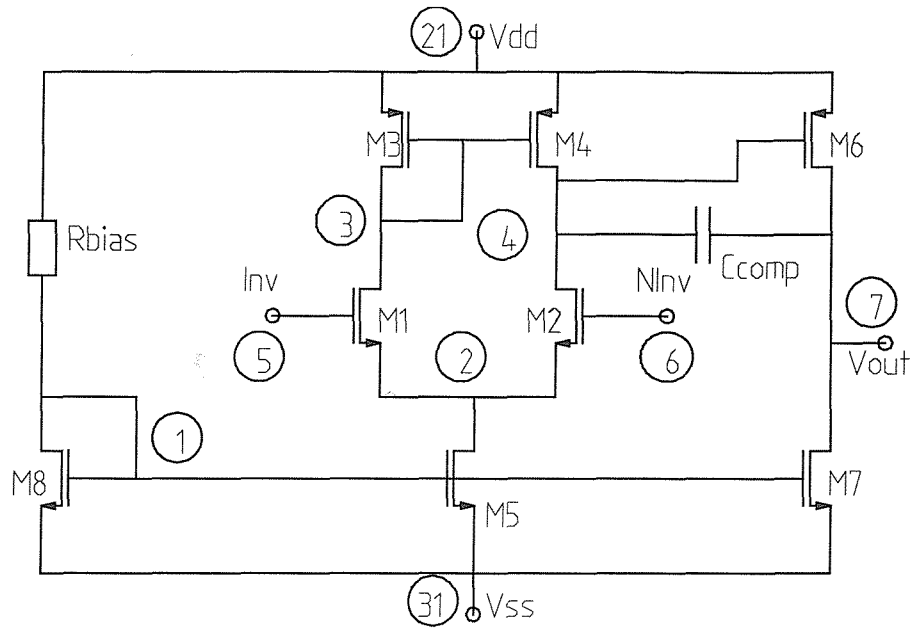


Figure 6-11. Two stage CMOS opamp used in [35] for behavioural fault modelling.

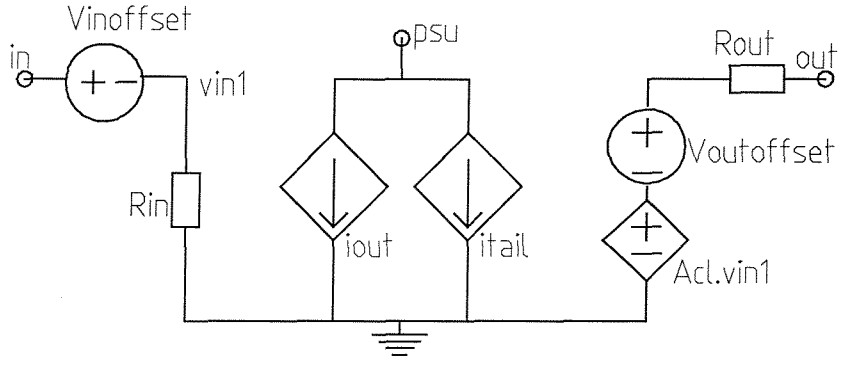


Figure 6-12. The macromodel used in [35] for the opamp of Figure 6-11 operating in closed-loop inverting amplifier configuration.

In order to group the inserted faults, a number of transistor level simulations including transient, AC and pole-zero analyses for the opamp configured as

inverting, non-inverting and summing amplifier were carried out [35]. As a result of fault grouping, up to 56% reduction in number of fault simulations for the opamp was reported in [35].

Complex circuits, such as an audio mixer circuit and a leapfrog filter, were also simulated in [35], where opamps used in those circuits were replaced with the behavioural equivalent obtained from the macromodel given in Figure 6-12. Note that while this macromodel includes the supply current, it does not capture stuck-at supply voltage fault effects, in contrast to macromodel given in Figure 6-10.

An effect coming from inserted faults on both the output voltage and the supply current for both circuits were observed by simulation at transistor and behavioural level [35]. As a result of these simulations, very similar results in terms of the output voltage and the supply current were obtained for the audio mixer circuit only. The CPU times required for fault simulations of two channels of the audio mixer circuit for 2ms with a stimulus of 1 kHz were compared for the three modelling approaches they used: a full transistor model; a SPICE macromodel and an analogue HDL model. It was reported that results obtained by the SPICE macromodel using HSPICE simulator were up to 4.8 times faster in terms of the CPU time than those obtained by the behavioural model written in HDL-A using ELDO simulator, where they were only up to 3.2 times faster than the results obtained using transistor level simulations with HSPICE simulator. One main reason why behavioural simulation with ELDO was slow is that the techniques implemented in ELDO to solve DAEs at that time were not as efficient as the numerical techniques implemented in HSPICE in terms of the simulation time.

The motivation for the rest of this chapter is, therefore, to investigate advances in the most recent analogue HDL simulators, while main emphasis is given

to behavioural fault modelling problem for analogue circuits. An analogue HDL, MAST [116], in conjunction with the SABER [116] simulator, and a VHDL-AMS simulator, Hamster [117], which is currently available free of charge, was used.

6.5.1 Behavioural fault model for the closed-loop inverting opamp in MAST

Let us consider the input-output relationship given in (6-1) under fault conditions for the benchmark opamp again. The faulty macromodel given in Figure 6-10 can now be simplified to that shown in Figure 6-13, where the opamp is assumed to be fault-free and ideal [107]. All the fault effects and non-ideal effects are approximated to $Fos = mV_{in} + k$, which is applied to the inverting input of the opamp.

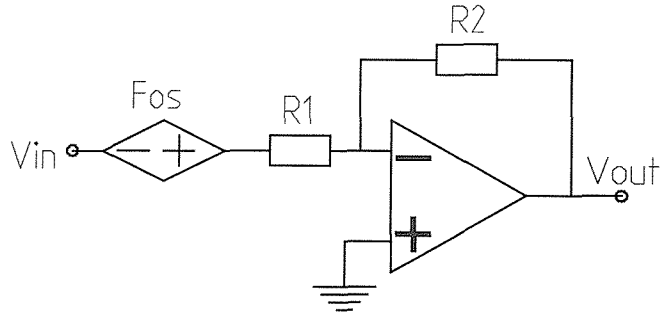


Figure 6-13. Behavioural level DC-offset fault model proposed in [107] for the inverting opamp.

A closed loop behavioural model given by (6-1) can be created using the MAST [116] modelling language, for the closed-loop operational amplifier. The basic model for the fault-free condition is given in Figure 6-14. Using dc-sweep analysis in SABER [116] simulator, the input-output relationship for the MAST behavioural model for the fault-free inverting opamp is shown in Figure 6-15.

```
Template opamp_behav2 vin vout
gnd=rin,rout,a,vosin,vosout,m,k
electrical vin,vout,gnd
#...Operational Amplifier Primitive Parameters
number    rin=100e6
number    Acl=-1 # inverting case
#number   Acl=2  # non-inverting case
#...Fault Offset Voltage Parameters
number    m=0
number    k=0
{
#...Declarations
var i i
val v vo,vi,fo,voutcalc
#...Procedural Expressions
values {
     #...Terminal Voltages
     o = v(vout) - v(gnd)
     vi = v(vin) - v(gnd)
     #...Fault Offset Voltage
     fo = m*vi + k
     voutcalc = Acl*(vi+fo)
     #...Supply Voltage Limit
     if (voutcalc > 2.5)  {
          voutcalc=2.5
     }
     if (voutcalc< -2.5) {
          voutcalc = -2.5
     }
}
equations {
     #...Fundamental Equations
     i(vout->gnd) += i
     vo = voutcalc }
}
```

Figure 6-14. Faulty behavioural closed-loop inverting opamp model in MAST.

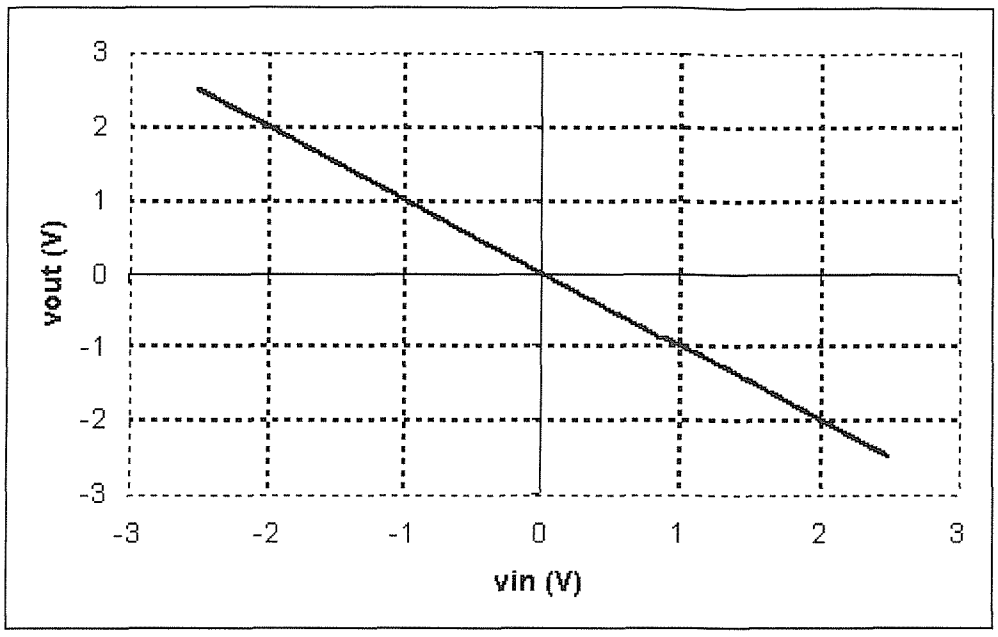


Figure 6-15. The input-output relationship found for the MAST model given in Figure 6-14 using SABER simulation (inverting opamp).

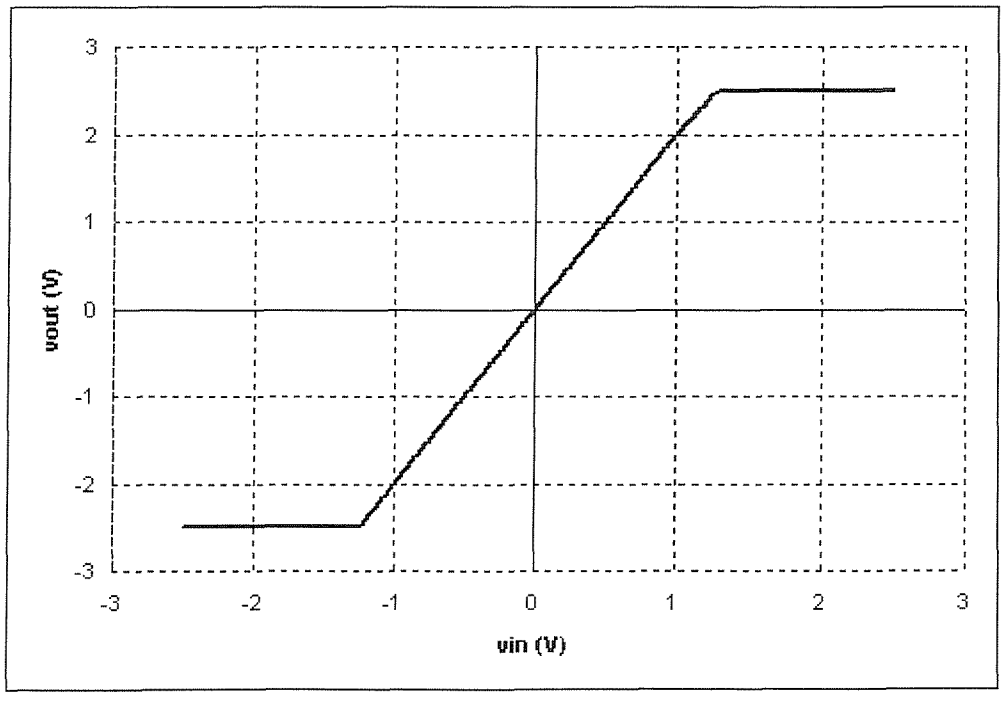


Figure 6-16. The input-output relationship found for the MAST model given in Figure 6-14 using SABER simulation (noninverting opamp).

The MAST model given in Figure 6-14 covers both inverting and noninverting behaviour of the opamp. The only difference is that A_{CL}=2 is used for noninverting case. A SABER simulation for the noninverting behavioural MAST model for the fault-free case is shown in Figure 6-16.

The behavioural model parameters *m* and *k* can be easily implemented in MAST and VHDL-AMS for different fault types (Type I to Type IV) using the equations given in section 6.3. One needs to first, however, determine the non-ideal effects, such as *Rid* and *Rdd*, occurring for each fault type by transistor level simulations. Another simpler way of determining these parameters is to approximate *m* and *k* such that they give similar responses to the ones found by transistor level simulations (i.e. curve fitting). The latter technique is easy to realise when the curve to be fitted is rather simple, such as for Type III faults (piecewise linear).

From the responses obtained by transistor level simulations (Figure 6-6 and Figure 6-7) one can approximate *m* = -1.02 and *k* = 2.15V for Type I faults, as these type of faults cause the output of the opamp to be nearly stuck-at some dc voltage level (~ -2.13V in the case given in Figure 6-6). Type II faults cause a small dc offset at the input of the opamp, hence selection of *m* = 0 and *k* = ~11mV provides a good correlation between the behavioural and transistor level simulations. For Type III and Type IV faults *m* and *k* can be determined as follows: if vin > 0V *m* = 0 and *k* = 0V, else *m* = -2 and *k* = 0V for Type III faults; and if vin > ~ 1.2V *m* = -1 and *k* = Vdd/2 (=1.25V), else if vin < ~ -1.2V *m* = -1 and *k* = Vss/2 (= -1.25V), else *m* = 0 and *k* = 0V for Type IV faults (note that $A_{CL} = 1 + \frac{R_2}{R_1} = 2$ for the opamp operating in the non-inverting amplifier configuration shown in Figure 6-2 (b)). The values of parameters *m* and *k* given above for different fault types are summarised in Table 6-I.

Table 6-I. The values of the parameters *m* and *k* for different fault types for the closed-loop inverting opamp behavioural model.

Parameters / Fault types	*m*	*k* [V]
Type I	-1.02	2.15
Type II	0	0.011
Type III	0 if vin > 0V -2 if vin < 0	0
Type IV	-1 if vin > ~1.2V and vin < ~ -1.2V 0 if ~ -1.2V < vin < ~ 1.2V	Vdd/2 if vin > ~ 1.V Vss/2 if vin < ~ -1.2V 0 if ~ -1.2V < vin < ~ 1.2V

Note that the parameters *m* and *k* can be realised using procedural statements in the behavioural model for Type III and Type IV faults. To do that, the MAST model given in Figure 6-14 was modified as shown in Figure 6-17 and Figure 6-18, with this code inserted in the procedural section of the model:

```
#...Fault Offset Voltage
if (vi < 0) {
     fo = -2*vi
}
else {
     fo = 0
}
vout = a*(vi+fo)
```

Figure 6-17. MAST implementation of *m* and *k* for Type III faults.

```
#...Fault Offset Voltage
if (vi > 1.2) {
     fo = -vi + 1.25
}
else if (vi < -1.2)
     fo = -vi - 1.25
}
else {
     fo = 0
}
vout = a*(vi+fo)
```

Figure 6-18. MAST implementation of *m* and *k* for Type IV faults.

The simulation results for different fault types using SABER simulator are shown in Figure 6-19, Figure 6-20, Figure 6-21, and Figure 6-22.

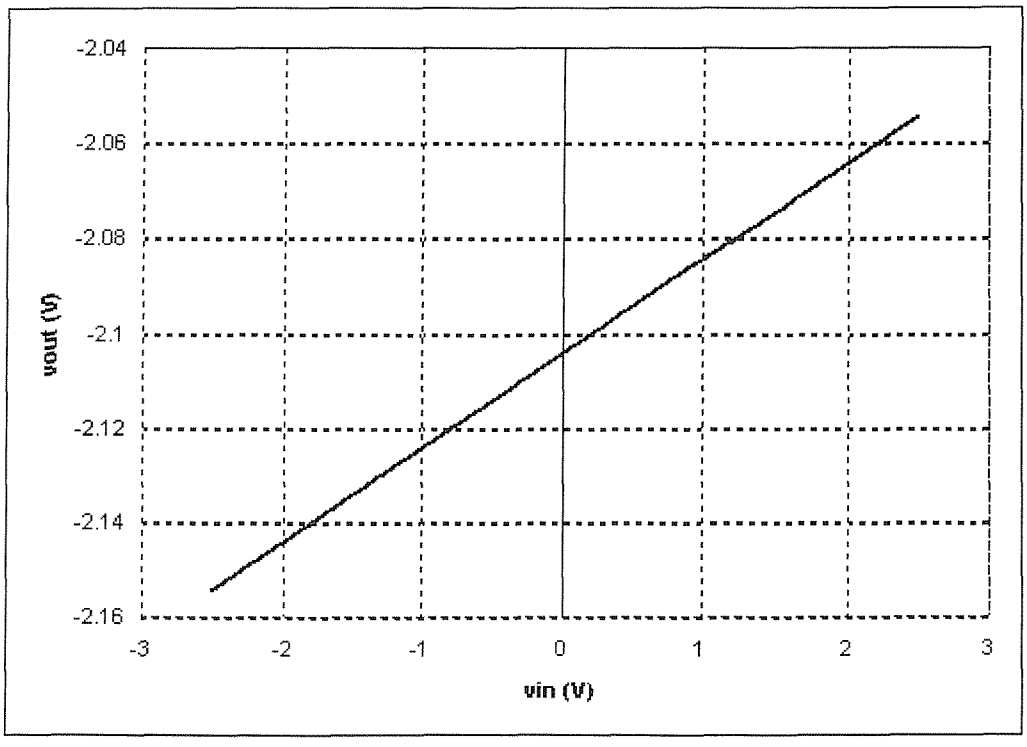


Figure 6-19. The input-output relationship found using SABER simulation for the MAST behavioural closed-loop opamp model for the Type I faults.

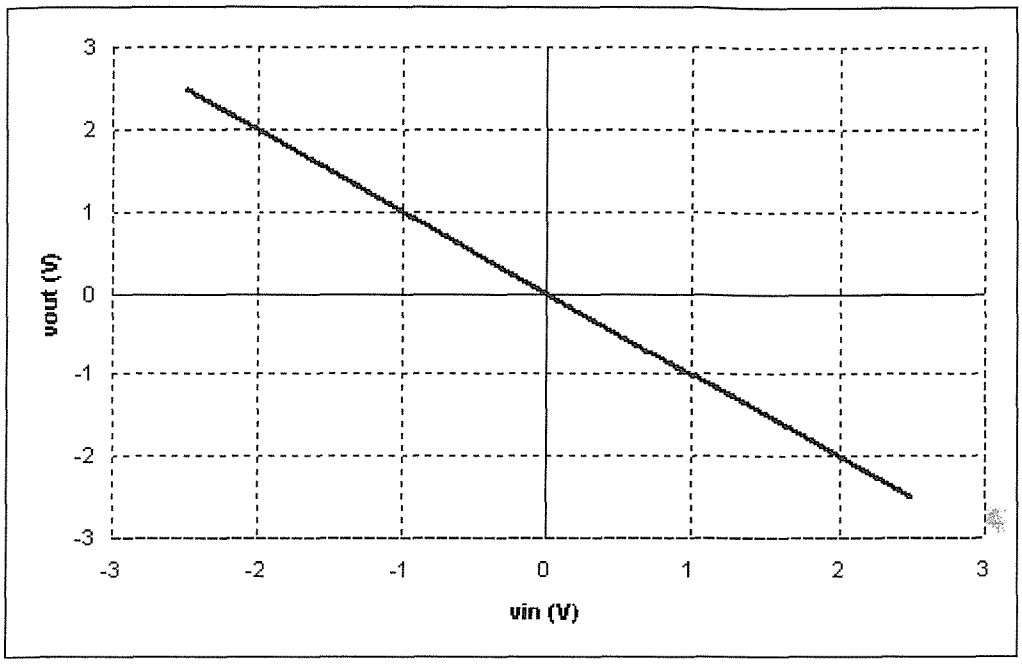


Figure 6-20. The input-output relationship found using SABER simulation for the MAST behavioural closed-loop opamp model for the Type II faults.

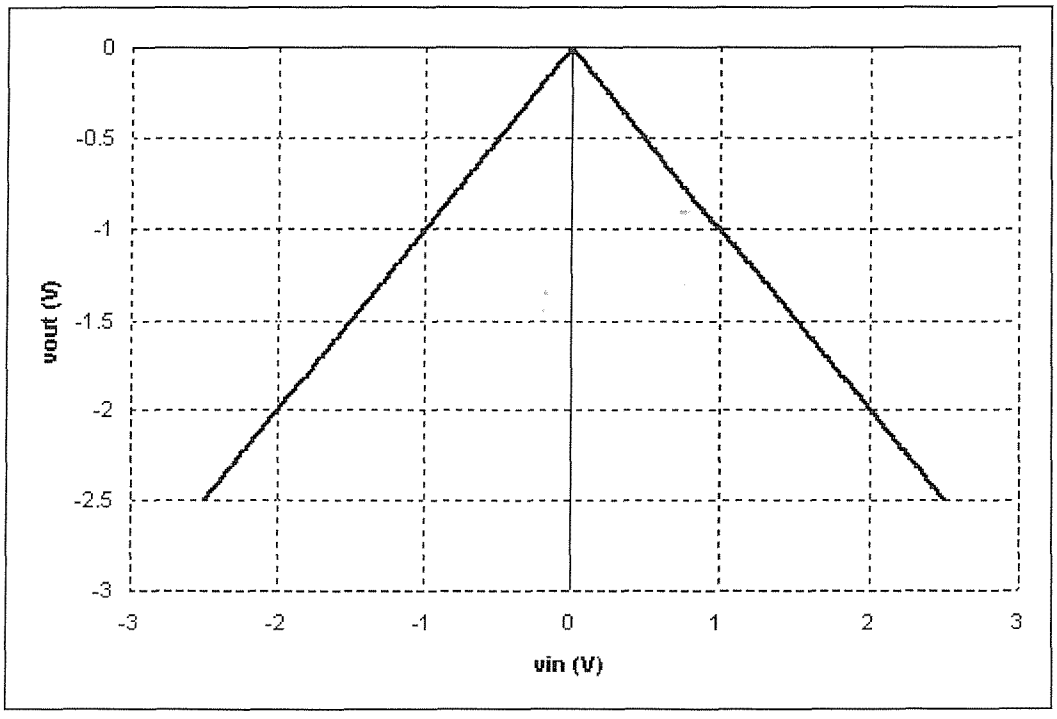


Figure 6-21. The input-output relationship found using SABER simulation for the MAST behavioural closed-loop opamp model for the Type III faults.

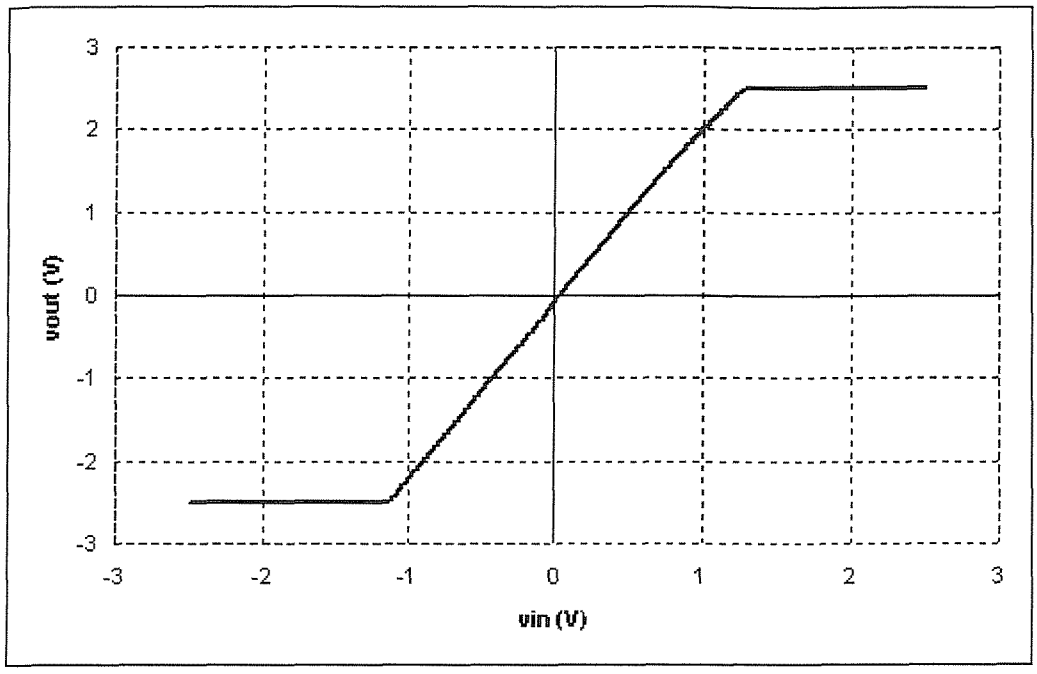


Figure 6-22. The input-output relationship found using SABER simulation for the MAST behavioural closed-loop opamp model for the Type IV faults.

In order to see the speed-up using the MAST behavioural models, DC sweep analysis simulations were carried out using SABER where both the MAST behavioural models and transistor level models were simulated. An input signal ranging from –2.5V to 2.5V was applied to the opamp's input with 0.01V step size for both cases. Table 6-II represents the comparison of CPU times for transistor level and the MAST behavioural level simulations carried out using SABER. As can be seen from the table the speed-up for type I and type II faults is 18 times where for type III faults it is around 12.7 and for type IV faults the speed-up is around 11.2. The reason why the speed-up is relatively smaller for type III and type IV faults is that it is needed to model those faults using procedural statements within the behavioural MAST model as shown in Figure 6-17 and Figure 6-18.

Table 6-II. Comparison of CPU times for different modelling approaches for DC-sweep analysis.

The CPU time (s) / Fault Type	Modelling Language	
	MAST	Transistor level
Fault I	0.1	1.8
Fault II	0.1	1.8
Fault III	0.15	1.9
Fault IV	0.17	1.9

6.5.2 Behavioural fault model for the closed-loop inverting opamp in VHDL-AMS

A VHDL-AMS implementation of the behavioural model given in (6-1) is shown in Figure 6-23, and Figure 6-24. As can be seen from the figures, it is much simpler to develop behavioural fault models using a standard analogue HDL (VHDL-AMS in this case) compared with the macromodel development using SPICE-like languages, such as [36] and [107]. In Figure 6-23, rin represents the input resistance of the opamp, where it is only used for the third equation in Figure 6-24. The third equation is needed as there are three quantities declared in the architecture declaration shown in Figure 6-24. Note that the architecture declaration given in Figure 6-24 also covers the supply voltage limiting effect at the output of the opamp.

```
--behavioural opamp
library disciplines;
library ieee;
use disciplines.electromagnetic_system.all;
use ieee.math_real.all;
--entity
entity op_behav is
   generic ( m : real := 0.0; --fault-free value
             k : real := 0.0; --fault-free value
             Acl : real := -1.0; --closed-loop gain
             rin : real := 100.0e6);
   port (terminal in_node,  out_node : electrical);
end;
```

Figure 6-23. The VHDL-AMS entity implementation of the behavioural fault model derived from Figure 6-10 for the inverting opamp.

```
--architecture
library disciplines;
library ieee;
use disciplines.electromagnetic_system.all;
use ieee.math_real.all;
architecture behav of op_behav is
   quantity vout across iout through out_node;
   quantity vin across iin through in_node;
   quantity Fos : real;
   -- supply voltage limit
   constant v_limit : real := 2.5;
begin
procedural is
  variable vout_calc : real;
  begin
   Fos := m*vin + k;
   vout_calc := Acl * (vin + Fos);
```

```
        iin  := (vin - Fos) / rin;
        if (vout_calc > v_limit) then vout := 2.5;
        elsif (vout_calc < -v_limit) then vout := -2.5;
        else vout := vout_calc;
        end if;
    end procedural;
    end;
```

Figure 6-24. The VHDL-AMS architecture implementation of the behavioural fault model derived from Figure 6-10 for the inverting opamp.

In order to simulate the VHDL-AMS model shown in Figure 6-23 and Figure 6-24, one also needs VHDL-AMS models for a resistor, a voltage source, and a testbench, which are shown in Figure 6-25, Figure 6-26, and Figure 6-27, respectively.

```
--resistor
library disciplines;
use disciplines.electromagnetic_system.all;
entity resistor is
  generic (rnom : real := 0.0);
  port(terminal p,m  : electrical); --interface ports.
end resistor;
architecture behav of resistor is
    quantity r_e across r_i through p to m;
begin
    r_i == r_e/rnom;
end behav;
```

Figure 6-25. A VHDL-AMS model of a resistor.

```
-- voltage source
library disciplines;
use disciplines.electromagnetic_system.all;
--entity declaration.
entity v_source is
    generic (dc_value : real :=-2.50);
     port(terminal p,m: electrical);--interface ports.
end v_source;
--architecture declaration.
architecture behav of v_source is
    quantity v_in across i_out through p to m;
begin
  v_in==dc_value*now; -- slow transient
end architecture behav;
```

Figure 6-26. A VHDL-AMS model of a voltage source.

Note that input voltage source in architecture declaration shown in Figure 6-26 is realised using a predefined VHDL-AMS function, *now*, which returns the value of the current time at each step as simulation proceeds. This is done in order to simulate the dc-sweep analysis, which is not defined in VHDL-AMS (contrary to SPICE-like languages). This technique is called *slow transient* simulation.

```
--testbench
library disciplines;
library ieee;
use disciplines.electromagnetic_system.all;
use ieee.math_real.all;
entity ex_op_behav is end;
architecture testbench of ex_op_behav is
  terminal in_node, out_node, vsrc : electrical;
begin
```

```
  op_behav_uut : entity op_behav (behav)
    generic map ( m => 0.0,k => 0.0, acl => -1.0)
    port map ( in_node  => in_node,
               out_node => out_node);
  vin_dc : entity v_source (behav)
  generic map ( dc_value => 5.0)
  port map ( p => vsrc, m => electrical_ground);
  rsrc : entity resistor (behav)
    generic map (rnom => 100.0)
    port map (p => vsrc, m => in_node);
end;
```

Figure 6-27. A VHDL-AMS testbench used with the Hamster simulator to simulate the behavioural model shown in Figure 6-23 and Figure 6-24.

The slow transient simulation results using the Hamster [117] simulator and the behavioural closed-loop VHDL-AMS model of the inverting opamp (the fault free case) with the necessary component and voltage source models and the testbench shown in Figure 6-23-Figure 6-27 are shown in Figure 6-28 and Figure 6-29 for both the positive and the negative values of the input voltage source. Note that the X-axis in in Figure 6-28 and Figure 6-29 represents the time in seconds, where Y-axis represents vout, vin, and Fos in Volts. (Unless otherwise stated, for the rest of this chapter it will be assumed that X-axis will represent time in seconds for the simulation results obtained using Hamster).

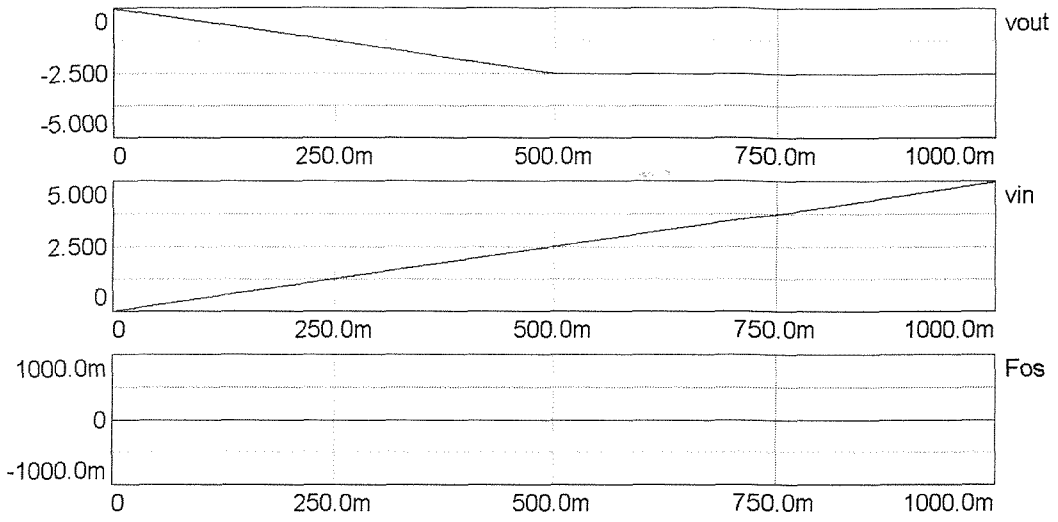


Figure 6-28. Slow-transient simulation results using the VHDL-AMS model with Hamster for the positive values of the input voltage source.

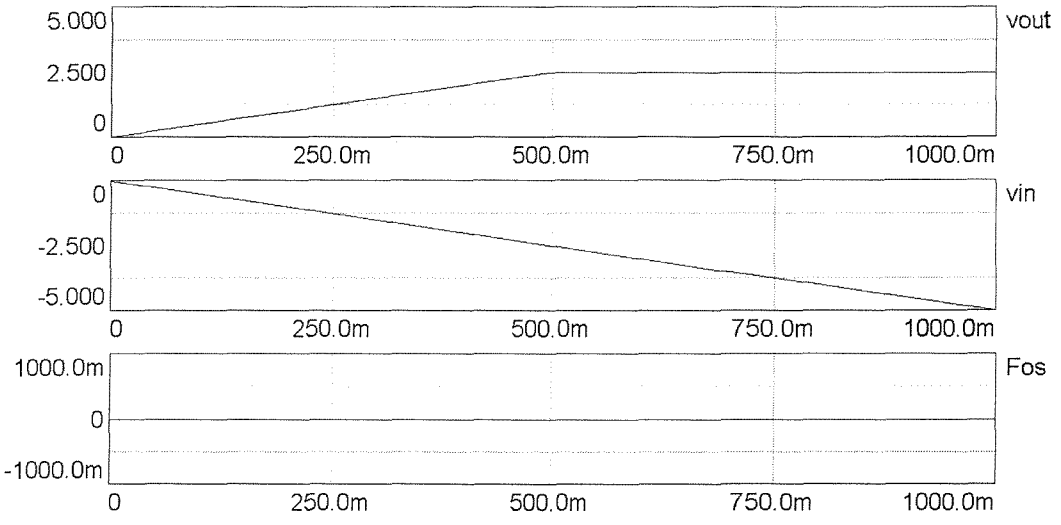


Figure 6-29. Slow-transient simulation results using the VHDL-AMS model with Hamster for the negative values of the input voltage source.

Let us consider the same values for the parameters m and k as those given in the previous section for different fault types (in Table 6-I). The simulation results using the Hamster for different fault types are shown in Figure 6-30, Figure 6-31, Figure 6-32, Figure 6-33, Figure 6-34, Figure 6-35, Figure 6-38, and Figure 6-39.

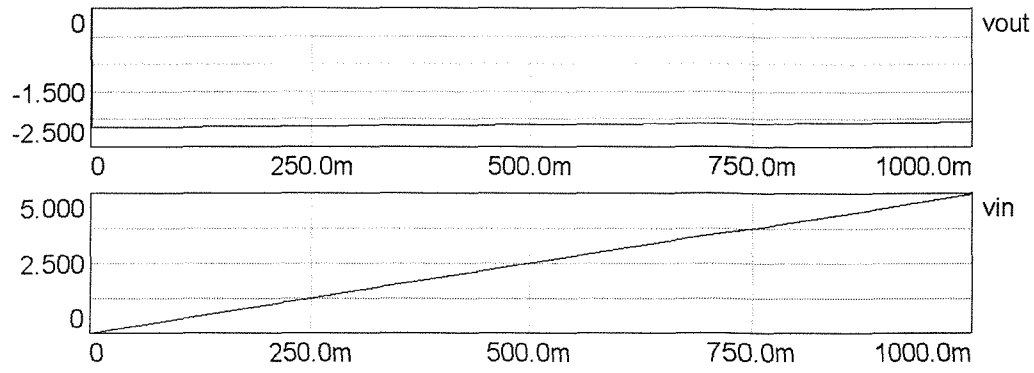


Figure 6-30. The VHDL-AMS behavioural closed-loop inverting opamp model slow-transient simulation using Hamster for Type I faults for the positive values of vin.

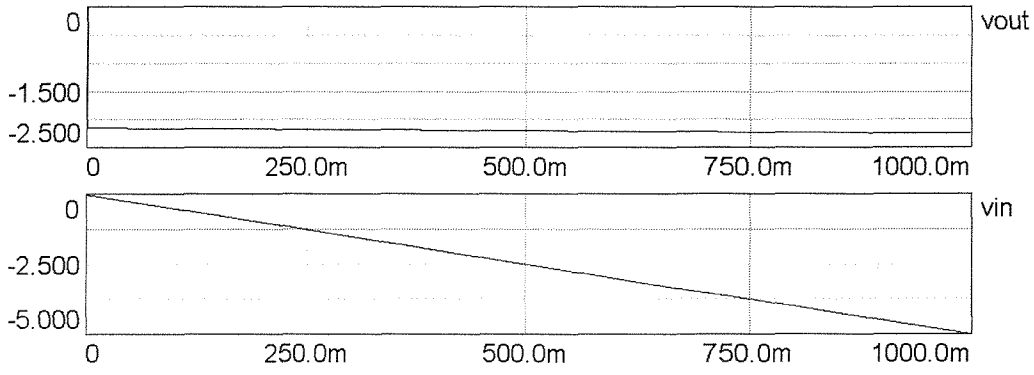


Figure 6-31. The VHDL-AMS behavioural closed-loop inverting opamp model slow-transient simulation using Hamster for Type I faults for the negative values of vin.

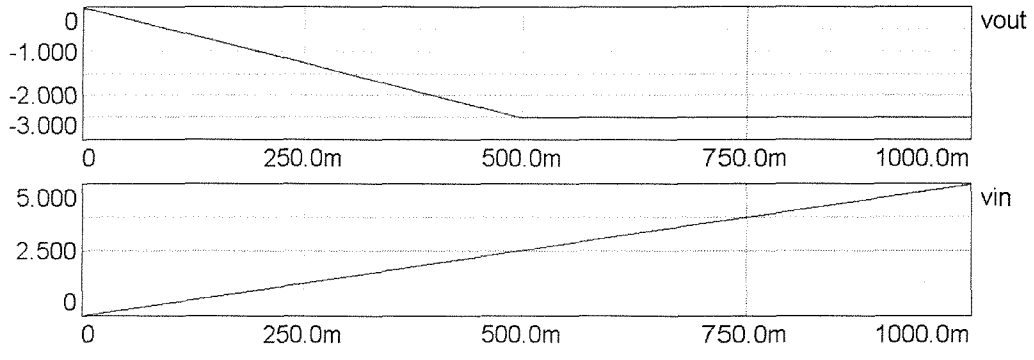


Figure 6-32. The VHDL-AMS behavioural closed-loop inverting opamp model slow-transient simulation using Hamster for Type II faults for the positive values of vin.

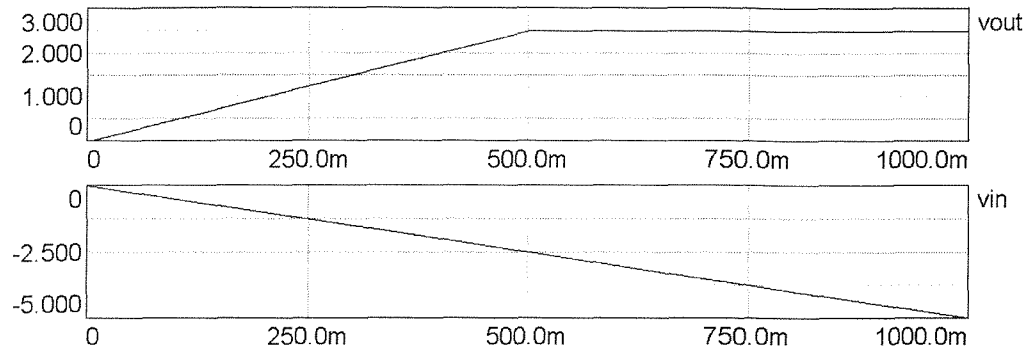


Figure 6-33. The VHDL-AMS behavioural closed-loop inverting opamp model slow-transient simulation using Hamster for Type II faults for the negative values of vin.

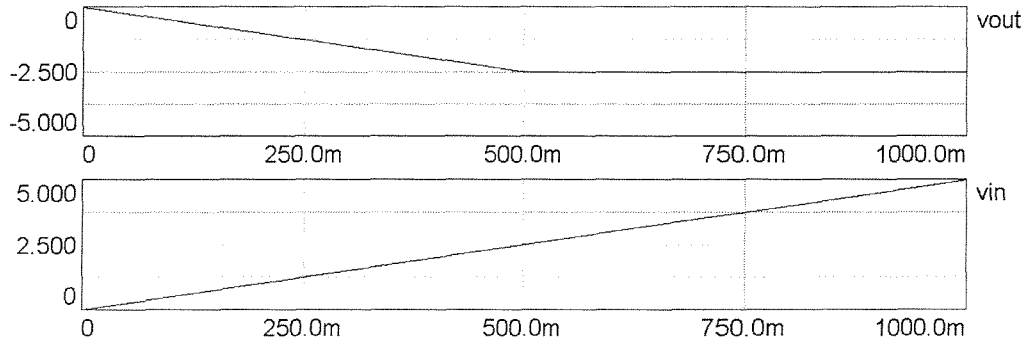


Figure 6-34. The VHDL-AMS behavioural closed-loop inverting opamp model slow-transient simulation using Hamster for Type III faults for the positive values of vin.

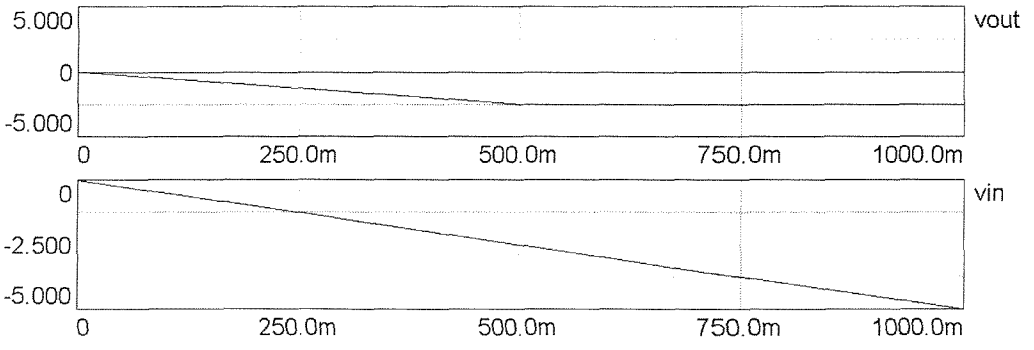


Figure 6-35. The VHDL-AMS behavioural closed-loop inverting opamp model slow-transient simulation using Hamster for Type III faults for the negative values of vin.

Note that the output response of the opamp, vout, found for the positive values of vin and the negative values of vin for Type I faults (Figure 6-30 and Figure 6-31) are the same (nearly stuck-at –2.14 V), as expected.

For Type III faults *Fos* is determined using the following if-then construct in the VHDL-AMS model:

```
if (vin < 0.0) then
  Fos := -2.0*vin;
else
  Fos := 0.0;
end if;
```

Figure 6-36. if-then construct implemented in the VHDL-AMS model for Type III faults.

For Type IV faults *Fos* is determined using the following if-then construct in the VHDL-AMS model:

```
if (vin > 1.2) then
   Fos := -vin + 1.25;
elsif (vin < -1.2) then
   Fos := -vin - 1.25;
else
   Fos := 0.0;
end if;
```

Figure 6-37. if-then construct implemented in the VHDL-AMS model for Type IV faults.

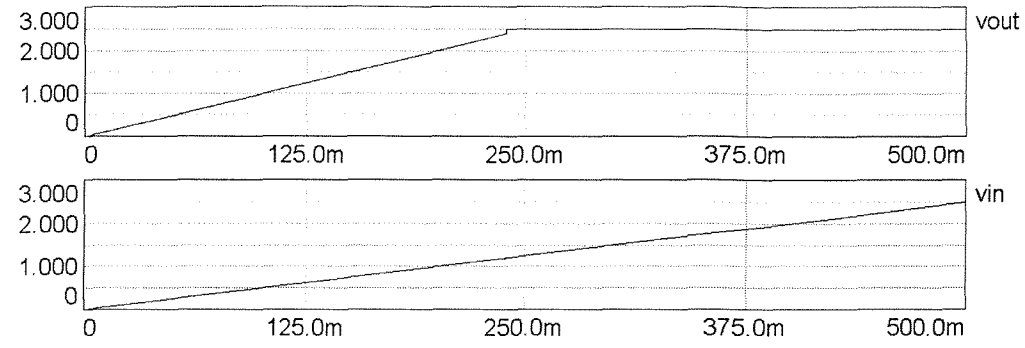


Figure 6-38. The VHDL-AMS behavioural closed-loop non-inverting opamp model slow-transient simulation using Hamster for Type IV faults for the positive values of vin.

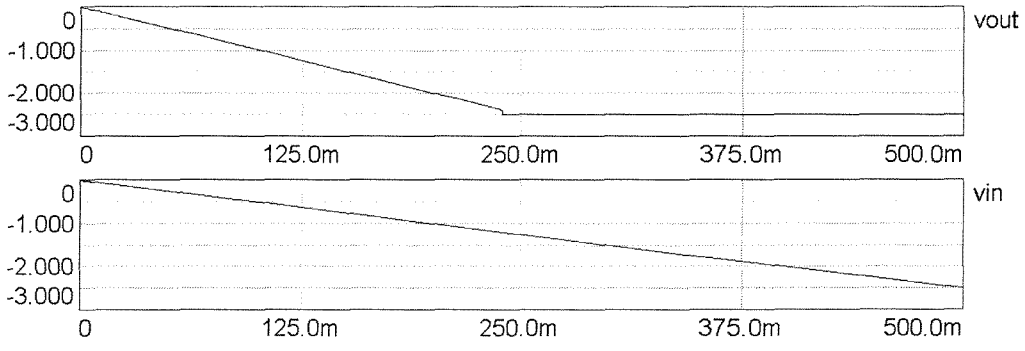


Figure 6-39. The VHDL-AMS behavioural closed-loop non-inverting opamp model slow-transient simulation using Hamster for Type IV faults for the negative values of vin.

As DC-sweep analysis cannot be performed for VHDL-AMS, the transient simulation results for different fault types using VHDL-AMS behavioural models and Hamster simulator were compared with the transient simulation results that are obtained using transistor level models with HSPICE simulator. To do that a sine wave with 2V peak-to-peak magnitude and 1KHz frequency was applied to both behavioural and transistor level circuits. The simulators were let to run for 5 ms with 10 μs iteration step. Table 6-III shows the CPU time spent for each case with the different approaches. As can be seen from the table there is an average of 4.4 times

speed-up for Fault I, Fault II cases. The speed-up for the type III faults between the behavioural and the transistor level simulations is 373.7 times, which is very large. The reason why the behavioural model is much faster than the transistor level is that type III faults are open drain faults where HSPICE needs more time to simulate that type of faults. Finally the speed-up for type IV faults is around 2.5 times. The behavioural model for type IV faults is relatively slower compared to other behavioural models in Table 6-III due to the procedural statement (Figure 6-37) needed to model the type IV faults.

Table 6-III. Comparison of CPU times for transistor level transient HSPICE simulations against VHDL-AMS behavioural level Hamster simulations.

The CPU time (s) / Fault Type	Simulator	
	HAMSTER	HSPICE
Fault I	90m	400m
Fault II	90m	360m
Fault III	100m	37.37
Fault IV	140m	350m

6.6 Conclusion

Analogue fault simulation is a key parameter to analogue/mixed-signal test generation. Currently such fault simulation is of limited use due to the speed of analogue simulation and the large number of faults to be simulated. Simulation can be speeded up by using number of techniques. Behavioural modelling is one of those

techniques. It has been shown in sections 6.5.1 and 6.5.2 how one can speed-up analogue fault simulation process by using behavioural models. Two analogue HDLs, namely, the MAST and the VHDL-AMS has been used. The advantage of the VHDL-AMS over the MAST and other proprietary analogue HDLs is that the VHDL-AMS is an IEEE standard. As VHDL-AMS was standardised in 1999, it is still in its infancy and there is not many VHDL-AMS simulators currently available. It is clear that by VHDL-AMS simulators getting more powerful it will be easier to model analogue/mixed-signal circuits at a higher level so as to speed-up simulation in general, analogue simulation in particular.

7 CONCLUSIONS

7.1 Summary

Faults in analogue circuits cannot be modelled as the digital single-stuck fault model. Typical fault models for analogue circuits include short and open circuits. Faults for analogue circuits are usually categorised as catastrophic and parametric faults. Catastrophic faults are the ones that affect the functionality of the analogue circuit dramatically whereas parametric faults cause slight variations in the functionality of the circuit. A number of testing strategies have been proposed for analogue circuits but are dependent on the choice of a good test stimulus.

Supply current monitoring is an effective method for testing digital and analogue circuits [6], [7]-[14]. Supply current monitoring can be implemented either using automatic test equipment (ATE) or Built-In Current Sensors (BICS). Using a BICS is more advantageous in terms of: test equipment costs; increasing the testing rate; improving the fault detectability and observability of the Circuit Under Test (CUT); higher current sensing resolution and avoiding the influence of I/O currents which may dominate the chip's total current [7]. Therefore, in this thesis different BICS approaches, particularly for analogue circuits, are investigated thoroughly in the Chapter 3. A new CMOS BICS circuit is developed and fabricated, which mainly

overcomes the drawback of process variation dependence of the previously published BICS circuits.

In many analogue CMOS circuits, most, if not all of the transistors are permanently in saturation, and cannot be switched by modifying the input stimulus. One way to cause transistors to change their normal mode of operation is to vary the supply voltage. In the Chapter 4, the effectiveness of the test technique based on varying the supply voltage, which tries to switch transistors to different mode of operation, in conjunction with monitoring the supply current has been shown through the use of a PLL circuit. Over 10% increase in the fault coverage of a complex analogue/mixed-signal circuit has been achieved using this technique. Even though with the voltage mode circuits (e.g. PLL) this technique does not give the desired fault coverage figures, it might do for today's low voltage and current mode circuits.

Fault simulation is the first step to fault coverage analysis, fault grading, fault collapsing, and BIST [28]. Fast fault simulation of analogue circuits is crucially important in terms of speeding up the analogue testing process. There are not many analogue fault simulation algorithms in contrast to digital fault simulation algorithms. This is mainly because there is not yet a standard fault model like stuck-at fault model of digital circuits available for analogue circuits. There exist quite a few techniques in order to achieve fast analogue fault simulation. Some of them are: fault dropping/collapsing, in which faults that cause similar changes in the circuit response compared with another faulty circuit response and/or with the fault-free circuit response are considered equivalent, hence dropped from further simulation. Therefore, in the Chapter 5, new techniques with regard to fault dropping are developed and implemented in C programming language and implemented into a SPICE-like analogue fault simulator (CAFS).

Another technique to speed up analogue fault simulation is to use behavioural models for the analogue circuit blocks within a complex analogue/mixed-signal circuit. In this technique, parts of the circuit are modelled at a more abstract level, therefore the complexity and the simulation time is reduced. Modelling the faulty analogue circuit at behavioural level is not a trivial task. Two main approaches to behavioural modelling are; analytical model, where the mathematical equations of the faulty behaviour are used to construct the behavioural model [96] and statistical behavioural modelling [109]. Hardware Description Languages (HDLs) have been successfully in use for behavioural modelling of digital circuits for a while now. Simulators that implement mixed-signal HDLs are developing rapidly. There is very little work, however, done [35] currently with limited success using analogue HDLs for behavioural fault modelling for analogue circuits. One main reason for this limited success is that the currently available analogue HDL simulators are not robust, which implement all the language construcs of analogue HDLs, as analogue HDLs became standard very recently (VHDL-AMS was standardised in 1999 for instance).

With the advent of mixed-mode HDLs such as VHDL-AMS it is easier to deal with behavioural modelling in general, behavioural fault modelling in particular of analogue and mixed-signal circuits, which will speed up the analogue fault simulation. Therefore, in the Chapter 6, we have shown how analogue fault simulation can be speeded up by using behavioural models that are implemented in analogue HDLs such as MAST and VHDL-AMS. The next section outlines the original contributions to this thesis.

7.2 Original Contributions of This Thesis

The original contributions of this thesis are as follows:

- A new built-in current sensor (BICS) circuit is designed and fabricated in 0.8μm AMS CYE CMOS (2.5-5.5V, p-sub, 2-metal, 2-poly) [15] technology. New BICS overcomes the main drawback of process variation independence with previously published sensors.
- An analogue test technique based on varying the supply voltage is applied to a complex CMOS analogue/mixed-signal circuit (PLL – Phase-Locked Loop). 10% overall fault coverage increase is obtained.
- New speed-up techniques allowing fault dropping for fault-based analogue fault simulation are developed in C programming language and implemented within a SPICE-like analogue fault simulator. Up to 100% fault coverage and up to 4.7 speed-up in terms of the CPU time is possible over the concurrent fault simulation with no fault dropping.
- Behavioural fault modelling in order to reduce analogue fault simulation time is investigated. Behavioural models using analogue HDLs such as MAST and VHDL-AMS have been developed. It has been shown that up to 373.7 times speed up in the CPU time is possible with the VHDL-AMS behavioural models, which are developed in the Chapter 6, over the transistor level models.

7.3 Recommendations for Further Work

The new process variation independent BICS circuit proposed in Chapter 3 of this thesis could be further investigated for its application to differential ended analogue circuits, and to digital circuits. It would also be useful to investigate how to integrate the proposed BICS design within an analogue/mixed-signal Built-In Self-Test (BIST) technique.

Supply voltage variation technique could be further investigated for today's low voltage, current-mode circuits where it could lead to better results in terms of fault coverage.

Fault dropping techniques are developed and are shown to increase the fault coverage and to reduce CPU time for fault simulation of analogue circuits. Fault collapsing can also be investigated, where the similarities between a faulty circuit and another faulty circuit could be exploited, in which case faults that are similar could be grouped.

As simulators that implement analogue HDLs are not mature yet, behavioural modelling using analogue HDLs for analogue circuits is still in its infancy. It has been shown in this thesis, however, how fault simulation can be speeded up using behavioural modelling approach with analogue HDLs. Further research on behavioural modelling using analogue HDLs will prove useful, as mixed-signal simulators such as Hamster are becoming available.

8 APPENDICES

8.1 C routines developed for CAFS

In Figure 8-1, automatic structural short faults list generation routine written in C for CAFS, which was described in Chapter 5 in detail, is given.

```
/* a routine for automatic fault list generation */

static int
ShortFaults (subcct *thiscct)
{
  nodevoltage *ptr1, *ptr2;
  long faultno=0, n=0, n_b=0, n_c=0;
  ptr1 = thiscct->nodelist;
  while (ptr1 != NULL)
  {
    if (ptr1->nodeno >= 0)
    {
      ptr2 = ptr1->nextnode;
      while (ptr2 != NULL)
      {
        if ((ptr2->nodeno >= 0) && (!ptr1->extnode ||
                                    !ptr2->extnode))
        {
          n++;
          thiscct->faults = newfault(thiscct->faults);
          thiscct->faults->ParamList =
                        newparam(thiscct->faults->ParamList);
          thiscct->faults->ParamList->pval.Value = 10.0;
          thiscct->faults->NodeList =
                     newdevicenode(thiscct->faults->NodeList);
          thiscct->faults->NodeList->NodeNo = ptr1->nodeno;
          thiscct->faults->NodeList->ActualNode = ptr1;
          thiscct->faults->NodeList->ExternalName =
                                      ptr1->NodeName;
```

```
            thiscct->faults->NodeList = newdevicenode(
                                thiscct->faults->NodeList);
            thiscct->faults->NodeList->NodeNo = ptr2->nodeno;
            thiscct->faults->NodeList->ActualNode = ptr2;
            thiscct->faults->NodeList->ExternalName =
                                              ptr2->NodeName;
          }
          ptr2 = ptr2->nextnode;
        }
      }
      ptr1 = ptr1->nextnode;
  }
  if (thiscct->child != NULL)
       n_c = ShortFaults (thiscct->child);
  else n_c = 0;
  if (thiscct->brother != NULL)
       n_b = ShortFaults(thiscct->brother);
  else n_b = 0;
  n += n_c;
  faultno = n + n_b;
  return faultno;
}
```

Figure 8-1. Automatic fault list generation routine in C.

In order to do closeness check between each faulty circuit response and the fault-free circuit response in the dc and transient analyses with CAFS the routines given in Figure 8-2 and Figure 8-3 are developed and implemented in C. The routine `Closeness(fptr, statfile)` used in Figure 8-2 and Figure 8-3, and given in Figure 8-4 is developed and implemented in C in order to calculate different user defined closeness measures.

```
/*the routine to carry out closeness check for the dc
analysis*/
void
checkDropDC (subcct *cct)
{
  subcct *fptr; fptr = cct;
  for (fptr = cct->brother; fptr; fptr = fptr->brother)
  {
    if (fptr->CurrentState==postactive)
    {
      fptr->dropList = newreal(fptr->dropList);
      fptr->dropList2 = newreal(fptr->dropList2);
      fptr->dropList->Value=0.0;
```

```
        fptr->dropList2->Value=0.0;
        fptr->dropList->Value = (double) Closeness(fptr,
                                  statfile);
        fprintf (statfile, "error_c = %f\n",
                  fptr->dropList->Value);
        fptr->dropList2->Value = (double) distance(fptr,
                                  statfile);
      fprintf (statfile, "error_d = %f\n",
                fptr->dropList2->Value);
    }}}
```

Figure 8-2. A routine to carry out closeness check for the dc analysis.

```
/*the routine to carry out closeness check for transient
analysis*/
void
checkDropTR (subcct *cct, double analysistime)
{
  subcct *fptr; fptr = cct;
  if (analysistime < analysistable.tstop && fptr->tLast ==
      analysistime && fptr->LastTime->TimePoint == fptr->
      friend_->LastTime->TimePoint)
 {
   fptr->dropList = newreal(fptr->dropList);
   fptr->dropList2 = newreal(fptr->dropList2);
   fptr->dropList->Value=0.0;
   fptr->dropList2->Value=0.0;
   fptr->dropList->Value = (double) Closeness(fptr, statfile);
   fprintf (statfile, "error_c = %f\n",
             fptr->dropList- >Value);
   fptr->dropList2->Value = (double) distance(fptr, statfile);
   fprintf (statfile, "error_d = %f\n",
             fptr->dropList2->Value);
 }}
```

Figure 8-3. A routine to carry out closeness check for transient analysis.

For the sake of simplicity closeness check is only given for single point multi node Euclidean distance in Figure 8-4, which was implemented in C for the DC analysis. The similar approach is adopted while calculating a distance measure for farness between a faulty circuit response and the fault-free circuit response. For transient analysis, the closeness measures are implemented in a very similar manner to the one implemented for the DC analysis.

In order to make use of these routines to calculate the distance measures for fault dropping purposes after the DC and during the transient analyses routines given in Figure 8-5 and Figure 8-6 are developed and implemented in C.

```
/* closeness check routine added */
float
Closeness(subcct *Faulty, FILE *statfile)
{
  nodevoltage *nodeFa, *nodeFF;
  subcct *FF;
  int M = 0;
  int n = 5 , i;
  float d_b, d_c, d = 0.0;
  FF = Faulty->friend_;
  FF->closeness = 0.0, FF->threshold = 0.0, FF->error = 0.0;
  nodeFa = Faulty->nodelist , nodeFF = FF->nodelist;

/*single point multi node Euclidean distance */
   if ((analysistable.seuclid || analysistable.meuclid) &&
       analysistable.mnodedc)
   {
      while (nodeFa && nodeFF)
      {
         M++;
       FF->closeness += pow(nodeFa->Vm - nodeFF->Vm, 2.0);
       FF->threshold += pow(nodeFF->Vm,2.0);
       nodeFF = nodeFF->nextnode;
       nodeFa = nodeFa->nextnode;
       }
      FF->closeness = sqrt(FF->closeness / (float)M);
      FF->threshold = (analysistable.close)*sqrt(FF->threshold
                      / (float)M);
      d = (FF->closeness - FF->threshold);
      fprintf(statfile,"Euclidean_sm = %f\t",FF->closeness);
      fprintf(statfile,"threshold = %f\t",FF->threshold);

      if (Faulty->child)
       d_c = ClosenessDC(Faulty->child, statfile);
      else d_c = 0.0;
       d += d_c;
      if (Faulty->brother && Faulty->parent)
          d_b = ClosenessDC(Faulty->brother, statfile);
      else d_b = 0.0;
      FF->error = (d + d_b);
      return FF->error;
   }
```

Figure 8-4. Closeness check routine.

```
/* the routine to drop faults after the DC analysis*/
void
dropFaultsDC (subcct *cct)
{
  int i, k, l, m, faults = 0;
  int d_faults_c = 0, d_faults_d = 0, t_faults = 0;
  double error_c, error_d;
  subcct *fptr;

  for (fptr = cct->brother; fptr; fptr = fptr->brother)
  {
    if (fptr->CurrentState == postactive)
    {
       faults++;
       error_c = 0.0, error_d = 0.0;
       i = 0, k = 0, l = 0, m = 0;
       while (fptr->dropList)
       {
         k++;
        error_c += fptr->dropList->Value;
         if (fptr->dropList->Value < 10E-4)
             i++;
         fptr->dropList = fptr->dropList->NextReal;
       }
       while (fptr->dropList2)
       {
         m++;
        error_d +=   fptr->dropList2->Value;
         if (fptr->dropList2->Value > 10E-4)
             l++;
         fptr->dropList2 = fptr->dropList2->NextReal;
       }
       if (k != 0 && m != 0)
       {
         fprintf (statfile, "\ner_c_sum = %10f",error_c);
         fprintf (statfile, " %3i time points dist < thr", i);
         fprintf (statfile, " of total %i time points\n", k);
         fprintf (statfile, "er_d_sum = %10f",error_d);
         fprintf (statfile, " %3i time points dist > thr", l);
         fprintf (statfile, " of total %i time points\n", m);
         fprintf (statfile, "-----------------------------");
         fprintf (statfile, "-----------------------------\n");
         if (error_c <= 10E-4)
        {
           d_faults_c++;
           fptr->CurrentState = dontanalyse;
        }
        if (error_d >= 10E-4)
        {
           d_faults_d++;
           fptr->CurrentState = dontanalyse;
        }
       }
       else
          fprintf (statfile, "\nWarning: no distance
```

```
                    calculation!\n");
      }
    }
    t_faults = d_faults_c + d_faults_d;
    fprintf (statfile, "\n*****************");
    fprintf (statfile, "******************\n");
    fprintf (statfile, "%i faults are simulated in total.\n",
             faults);
    fprintf (statfile, "\n%i faults are dropped at DC
             analysis.", t_faults);
    fprintf (statfile, "\n(%i close faults and %i far
             faults)\n", d_faults_c, d_faults_d);
    fprintf (statfile, "******************");
    fprintf (statfile, "******************\n\n");
}
```

Figure 8-5. A routine to drop faults after the DC analysis.

```
/*the routine added to drop faults for transient analysis */
void
dropFaultsTR (subcct *cct)
{
  int i, k, l, m, faults = 0;
  int d_faults_c = 0, d_faults_d = 0, t_faults = 0;
  double error_c, error_d;
  subcct *fptr;

  for (fptr = cct->brother; fptr; fptr = fptr->brother)
  {
    if (fptr->dropList && fptr->dropList2)
    {
       faults++;
       error_c = 0.0, error_d = 0.0;
       i = 0, k = 0, l = 0, m = 0;
       while (fptr->dropList)
       {
         k++;
         error_c += fptr->dropList->Value;
         if (fptr->dropList->Value < 10E-4)
             i++;
         fptr->dropList = fptr->dropList->NextReal;
       }
       while (fptr->dropList2)
       {
         m++;
        error_d +=   fptr->dropList2->Value;
         if (fptr->dropList2->Value > 10E-4)
             l++;
         fptr->dropList2 = fptr->dropList2->NextReal;
       }
       if (k != 0 && m != 0)
       {
         fprintf (statfile, "\ner_c_sum = %10f",error_c);
```

```
            fprintf (statfile, " %3i time points dist < thr", i);
            fprintf (statfile, " of total %i time points\n", k);
            fprintf (statfile, "er_d_sum = %10f",error_d);
            fprintf (statfile, " %3i time points dist > thr", l);
            fprintf (statfile, " of total %i time points\n", m);
            fprintf (statfile, "------------------------------");
            fprintf (statfile, "------------------------------\n");
            if (error_c <= 10E-4)
            {
              d_faults_c++;
              fptr->CurrentState = dontanalyse;
            }
            if (error_d >= 10E-4)
            {
              d_faults_d++;
              fptr->CurrentState = dontanalyse;
            }
          }
          else
             fprintf (statfile, "\nWarning: no distance
                      calculation!\n");
      }
    }
    t_faults = d_faults_c + d_faults_d;
    fprintf (statfile, "\n************************");
    fprintf (statfile, "************************\n");
    fprintf (statfile, "%i faults are considered at TR fault
             simulation.\n", faults);
    fprintf (statfile, "\n%i faults in total are dropped at TR
             analysis.", t_faults);
    fprintf (statfile, "\n(%i close faults and %i far
             faults)\n", d_faults_c, d_faults_d);
    fprintf (statfile, "************************");
    fprintf (statfile, "************************\n\n");
}
```

Figure 8-6. A routine to drop faults during the transient analysis.

8.2 VHDL-AMS

As VHDL-AMS is now an IEEE standard, in the next section a brief introduction to VHDL-AMS is presented. The VHDL-AMS is intended to cover a very large application area. Therefore, the discussion here will be limited to the most relevant parts of VHDL-AMS to this thesis, particularly to analogue behavioural

modelling using VHDL-AMS. More detailed information about VHDL-AMS can be found in [86], [118], and [119].

The IEEE standard 1076.1, also known as VHDL-AMS, is a hardware description language supporting the description and simulation of digital, analogue, and mixed analogue/digital (often called mixed-signal) systems in a single language.

8.2.1 VHDL-AMS Architecture

The VHDL-AMS language architecture as presented in [120] is given in Figure 8-7. As can be seen from the Figure 8-7, VHDL-AMS is not a new language. It is based on VHDL 1076-1993. It supports all VHDL 1076-1993 syntax and semantics.

VHDL-AMS adds new simulation models that support continuous behaviour. Continuous models are based on differential and algebraic equations (DAEs). DAEs are solved by a dedicated simulation kernel, the *analogue solver*. VHDL-AMS handles initial conditions, piecewise-defined behaviour, and discontinuities. Optimisation of the set of DAEs to be solved and how the analogue solver computes its solution are outside the scope of VHDL-AMS [120].

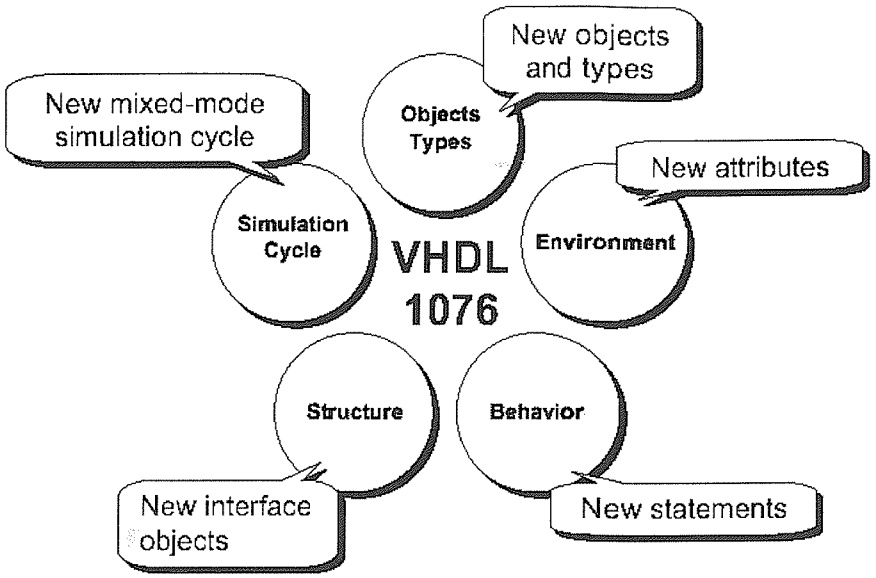


Figure 8-7. VHDL-AMS Language Architecture [120].

8.2.2 Differential and Algebraic Equations

The continuous aspects of the behaviour of the lumped systems targeted by VHDL-AMS can be described by a system of ordinary DAEs of the form given in [86] as

$$\mathbf{f}(x,\ x',\ t)=0 \qquad (8\text{-}1)$$

where $\mathbf{f}$ is a vector of expressions, x is a vector of unknowns, x' is a vector of derivatives of the unknowns with respect to time, and t represents time. Generally such systems of equations have no analytic solution [85]. Therefore, the solution must be approximated using numerical techniques.

8.3 Language Elements of VHDL-AMS

In this section the language elements of VHDL-AMS for the description of continuous time and mixed continuous/discrete time systems are summarised mostly from [86].

8.3.1 Overview of VHDL-AMS Models

A VHDL-AMS model consists of an *entity* and one or more *architectures.* The entity specifies the interface of the model to the outside world. It includes the description of the *ports* of the model (the points that can be connected to other models) and the definition of its *generic* parameters. The architecture contains the implementation of the model. It may be coded using a structural style of description, a behavioural style, or a style combining structural and behavioural elements.

A structural description is a netlist; it is a hierarchical decomposition of the model into appropriately connected instances of other models. A behavioural description consists of concurrent statements to describe event-driven behaviour and simultaneous statements to describe continuous behaviour. Concurrent statements include the concurrent signal assignment for data flow modelling and the process statement for more general event-driven modelling. Simultaneous statements are discussed in a later section.

When a VHDL-AMS model is instantiated in a structural description, the designer can specify which of several architectures to use for each instance. Alternatively, the decision can be postponed until immediately prior to the simulation. This allows for an easy and flexible reconfiguration of the model. For example, in top-down design, one architecture can describe a subsystem behaviourally with little detail, while another can add parasitics and a third can decompose the subsystem into lower level components.

8.3.2 Quantities

The unknowns in the collection of DAEs implied by the text of a model are analytic functions of time; that is, they are piecewise continuous with a finite number

of discontinuities. The analogue solver solves for the values of all unknowns over time by first converting the differential part of the DAEs, at specific values of time, to algebraic equations using appropriate discretization methods, and then solving the algebraic equations simultaneously.

VHDL-AMS introduces a new class of objects, the *quantity*, to represent the unknowns in the DAEs. Quantities can be scalar or composite (arrays and records), but must have scalar subelements of a floating-point type. A quantity object can appear anywhere a value of the type is allowed, in particular in an expression. In the rest of this section the characteristics of scalar quantities are described. The characteristics of a composite quantity are simply the aggregation of the characteristics of its scalar subelements. The behaviour of each scalar subelement is independent of the others.

The following statement declares three quantities q1, q2, and q3 of type REAL:

```
quantity q1, q2, q3: REAL;
```

where bold text indicates reserved words and upper-case text indicates predefined concepts.

A quantity can also be declared as an interface element in a port list of a model. Interface quantities support signal flow modelling. Each has a mode, similar in concept to the mode of an interface signal, indicating the direction of signal flow. For example, Figure 8-8 shows the entity declaration of a signal flow model with two interface quantities of mode **in** and one interface quantity of mode **out**.

```
entity summer is
    port (quantity in1, in2: in REAL;
          quantity sum: out REAL);
end entity summer;
```

Figure 8-8. Entity declaration of a signal-flow model.

When this model is instantiated each interface quantity is associated with a quantity declared in the instantiating model. For example

```
a1: entity summer port map (in1 => q1,
    in2 => q2, sum => q3);
```

The effect of a quantity association is to constrain the two quantities to be equal.

In addition to quantities declared explicitly, some quantities are implicitly declared by using their name in the text of a model. For example, Q'Dot is a quantity that holds the derivative of quantity Q with respect to time. Implicit quantities also exist for the integral of a quantity over time, ideal delay, Laplace, z-domain transfer functions, and ideal sample-and-hold. Other implicit quantities with regard to conservative systems modelling are described in another section.

8.3.3 Simultaneous Statements

Simultaneous statements are a new class of statements in VHDL-AMS for notating differential and algebraic equations. Simultaneous statements can appear anywhere a concurrent signal assignment is allowed. The basic form is the *simple simultaneous statement*, which has the following syntax:

```
[label:] expression == expression        (8-2)
```

where the square brackets indicate that this part of the statement is optional. For instance, the constitutive equation of a signal flow model relating to Figure 8-8 could be written as `sum == in1 + in2`.

The expressions may have composite values, in which case there must be a matching subelement on the left for each subelement on the right. The expressions may refer to signals, quantities, constants, literals, and functions. Each scalar subelement of a simple simultaneous statement is mapped to one expression in the vector **f** in (8-1) by subtracting the right expression in (8-2) from the left expression. When the analogue solver has properly established the value of each quantity, the matching subelements of the expressions will be approximately equal.

Several additional forms of the simultaneous statement have been defined. The *simultaneous case statement* and *simultaneous if statement* are analogous to their sequential counterparts and allow the description of piecewise defined behaviour. Each contains an arbitrary list of simultaneous statements in its statement parts, including nested simultaneous case and if statements. The analogue solver considers only the simultaneous statements selected by the case expressions and chosen by the conditional expressions. The *simultaneous procedural statement* allows local variables to be used, which is very important for modelling.

8.3.4 Conservative Systems

Systems with conservation semantics, such as electrical systems obeying Kirchhoff's laws, merit separate treatment as they are so commonly encountered. Special purpose syntax and semantics can provide a simplified notation, which reduces the risk of errors and thereby improves productivity. In VHDL-AMS, the

modeller need not explicitly notate equations describing the conservative aspects of such a system. Only the constitutive equations remain the modeller's responsibility.

The description of conservative systems uses a graph-based conceptual model. Let us consider the bipolar inverter circuit and its equivalent graph, shown in Figure 8-9. The vertices of the graph represent equipotential nodes in the circuit, and the edges represent branches of the circuit through which current flows.

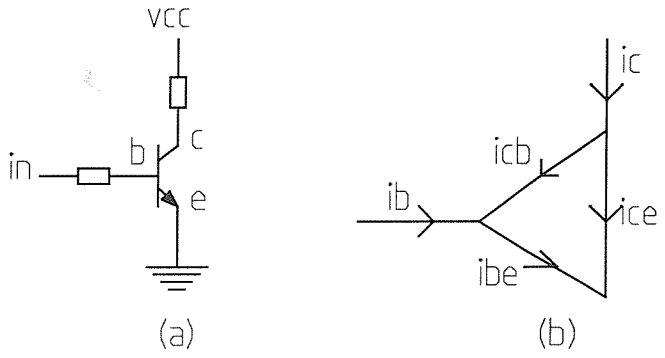


Figure 8-9. (a) Bipolar inverter and (b) its equivalent graph [86].

Similar graphs can be created for systems in other energy domains such as the thermal or fluidic domains. This conceptual model does not dictate any particular implementation for the analogue solver. In particular, it does not force the selection of a "nodal" formulation technique, e.g. the modified nodal formulation for instance.

Branch quantities represent the unknowns in the equations describing conservative systems. There are two kinds of branch quantities: *across quantities* and *through quantities*. Across quantities represent effort-like effects such as voltage, temperature, or pressure. They correspond to the potential difference between two vertices in the graph. Through quantities represent flow-like effects such as current, heat flow rate, or fluid flow rate. They correspond to the edges in the graph. The constitutive equations of conservative systems are expressed by relating the across

and through quantities of one or several branches using simultaneous statements. For example, a resistor has a single branch, and its constitutive equation (Ohm's law) relates the voltage across (the across quantity) and the current through (the through quantity) the resistor: i = v/r (nodal analysis).

A branch quantity is declared with reference to two *terminals*. The terminal is the second new object of the extended language. A terminal is declared to be of some *nature*. Natures can be scalar or composite (arrays and records). Each scalar nature represents a distinct energy domain, such as electrical, thermal, fluidic, etc. Its definition includes the types of across and through quantities incident to a terminal of the nature, and the common *reference terminal* (e.g., electrical ground, or mechanical anchor) shared by all terminals with elements of that scalar nature. These concepts are given in Figure 8-10. In the figure the declarations of two subtypes voltage and current, and a scalar nature electrical are shown. For reuse, these declarations are contained in a package electrical_system.

```
package electrical_system is
    subtype voltage is REAL;
    subtype current is REAL;
    nature electrical is voltage across
    current through ground reference;
end package electrical_system;
```

Figure 8-10. Declaration of nature electrical in package electrical_system [86].

Using the declarations given in Figure 8-10, the following statements declare two terminals t1 and t2 of nature electrical, and across quantity v, and two through quantities i1 and i2 between the terminals:

```
terminal t1, t2: electrical;
quantity v across i1, i2 through t1 to t2;
```

The across quantity represents the potential difference between the terminals and the through quantities, i1 and i2, represent two parallel current-carrying branches. The type of a branch quantity is not explicitly declared. Rather, it is derived from the nature of its terminals. It may be a composite type. In the example the across quantity v is of type voltage, and the type of the two through quantities i1 and i2 is current. As in the case of ordinary quantities, the characteristics of the composite are just the aggregate of the characteristics of its scalar subelements. The terminals must have elements of the same scalar nature. The terminals of a branch quantity are called the plus terminal and minus terminal, and the direction of the branch is plus to minus (in an electrical system), the direction of positive current flow.

A terminal may be declared anywhere a signal declaration is allowed. In particular, a terminal can be an interface element in a port list. For instance, the following statement declares the interface of a diode:

```
port (terminal anode, cathode : electrical);
```

When a model is instantiated, the association of interface terminals is used to construct nodes in hierarchical descriptions in a fashion paralleling the use of interface signals to construct nets in digital hierarchies.

The conservation equations of the system (in an electrical system, those equations due to Kirchhoff's laws) are extracted from the graph created by the

declared branch quantities and terminals and the association of terminals into nodes. A *node* is a set of scalar terminals created by a tree of terminal associations. The value of each scalar across quantity is constrained to be equal to the difference of the reference quantities of its terminals. All the reference quantities of the terminals of a node are constrained to be equal, and the contribution quantity of the terminal at the root of the tree is constrained to zero.

With these definitions a model of an ideal diode can be written as shown in Figure 8-11.

```
library ieee, disciplines;
use ieee.math_real.all;
use disciplines.electrical_system.all;
entity diode is
  generic (iss: real := 1.0e-14; -- saturation current
      af: real := 1.0; -- flicker noise coefficient
           kf: real := 0.0);-- flicker noise exponent
  port (terminal anode, cathode: electrical);
end entity diode;
architecture ideal of diode is
  quantity v across i through anode to cathode;
  constant vt: real := 0.0258; --thermal voltage at 300K
begin
     i == iss * (exp(v/vt) - 1.0);
end architecture ideal;
```

Figure 8-11. VHDL-AMS model of an ideal diode [86].

The library clause and the use clause make all declarations in the packages math_real and electrical_system visible in the model. This is necessary, because the model uses nature electrical from package electrical_system and function exp from package math_real. The entity declares the saturation current iss and the flicker noise

parameters af and kf as generics (i.e., parameters) and gives them default values, and anode and cathode as two interface terminals of nature electrical. The architecture declares v as across quantity and i as through quantity between anode and cathode, and a constant vt representing the thermal voltage. It includes a simple simultaneous statement describing the behaviour of the ideal diode.

8.4 Publications

8.4.1 Y. Kılıç and M. Zwolinski, "Testing Analog Circuits by Supply Voltage Variation and Supply Current Monitoring", The 1999 IEEE Custom Integrated Circuits Conference, May 16-19,1999, Town and Country Hotel, San Diego, California, USA.

Testing Analog Circuits by Supply Voltage Variation and Supply Current Monitoring

Y. Kılıç and M. Zwoliński

Department of Electronics and Computer Science
University of Southampton
Southampton SO17 1BJ, UK
{yk98r, mz}@ecs.soton.ac.uk

Abstract

A technique for sensitizing faults in analog circuits by varying the supply voltage is discussed. Unlike previous work, the technique is applied to the detection of short circuit faults. The validity of the technique is demonstrated with a simple CMOS circuit. The technique is applied to a larger analog circuit and significantly improved fault cover results are obtained.

Introduction

Testing of digital circuits has traditionally been based on the single-stuck fault model (although, of course, other fault models have been used). Test pattern generation algorithms based on this model attempt to force the node under consideration to the value opposite to that at which it might be stuck. The principle of *toggle testing* was proposed in which each node of a circuit is toggled between logic 0 and logic 1. IDDQ testing refined this idea such that the nodes of a circuit are switched to a logic value and under fault-free conditions the quiescent current is ideally negligible. Underlying all these techniques is the idea that individual transistors can be switched between conducting and non-conducting states.

Analog circuit testing, on the other hand, is much less structured. It is generally accepted that faults in analog circuits can be characterized by open and short circuits. The effects of such faults can be observed through the output voltages and supply currents, using DC, AC and other computed values and determining the deviation of such values from the nominal values. These measurements are made more difficult by the fact that parametric variations may cause the behavior of fault-free circuits to deviate significantly from nominal values. Hence the effect of a fault may be masked by its falling within the normal range.

In a manner analogous to that of digital circuits, the testability of analog circuits could theoretically be much improved if it were possible to cause transistors in the circuit to switch between different regions of operation. Under normal operating conditions, most of the transistors in a CMOS analog circuit are likely to be in saturation. If it were possible to switch some or all of the transistors to operate in the cutoff or triode regions, the difference between the fault-free and faulty behaviors would be likely to be very marked. It is possible to incorporate Design for Test structures that would give greater control over an analog block, but it is likely that such structures would be as complex as the original circuit.

Some control over the behavior of transistors within an analog circuit can be achieved by varying the supply voltage in conjunction with the inputs. Bruls used this idea to test a class AB amplifier at various supply voltage levels (1). He used the inductive fault analysis technique to insert processing defects into the layout of the IC in a random manner. A.K.B. A'ain, A.H.Bratt and A.P. Dorey applied a ramped power supply voltage to test small opamp circuits, and for exposing floating gate defects in analog CMOS circuits (2,3). The same authors applied an AC supply voltage to analog CMOS circuits (4). They achieved high fault covers with these tests, although the sizes of the circuits and the numbers of faults were small.

In this paper we show how varying the supply voltage of an analog circuit block can increase the fault cover of that block. Unlike the work described previously, we use a larger circuit element – a phase locked loop – and we model short circuit faults, which are much more likely to occur than open faults.

Transistor Switching

A typical, if simple, analog CMOS circuit is shown in Fig. 1. Under normal operating conditions, all of the transistors are in saturation all of the time. Therefore the quiescent current is naturally large (e.g. 300μA). Hence, the fault cover obtained by simply measuring the DC supply current is low. Other supply current measurements – the RMS value of the AC component and transient measurements can give a significantly higher fault cover, but such measurements must take into account the process parameter variations, which can make the distinguishing of faulty from fault-free behavior difficult.

When the opamp circuit of Fig. 1 is connected as an inverting amplifier, and operating linearly, all the transistors are in the saturation region, as noted. As the amplitude of the input is increased, various transistors start to operate in the MOS linear region. In particular as the input (in1) becomes more positive, M7 and M4 operate in the linear region. Similarly as in1 becomes more negative, M6 operates in the linear region. With still larger negative amplitudes, M2 also starts to operate in the linear region. Finally with a very large negative input, both M6 and M2 move into the cutoff region.

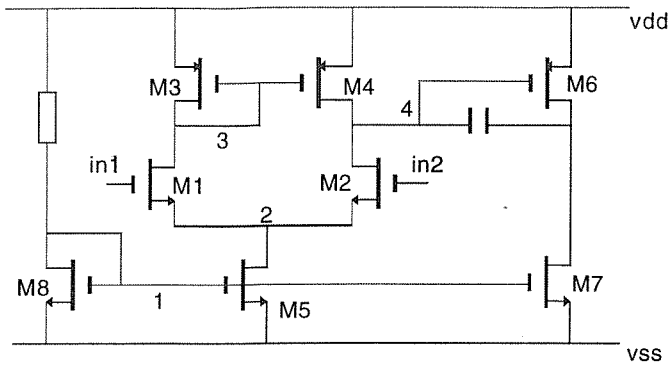


Figure 1 CMOS Opamp Circuit

The same effect can be achieved by varying the supply voltage. If VDD is reduced from the normal operating range of 5V to 4V or even 3V, M7 and M4 operate in the linear region and M6 can be forced into the cutoff region. A similar, but less pronounced effect can be achieved by varying VSS. Under these conditions, the opamp is itself in saturation. Hence the supply current is also saturated.

Under fault conditions, the advantages of varying the supply voltage become much more marked. A short circuit between the gate and drain of M4 was modeled by a 100Ω resistance between nodes 3 and 4. With this fault, the opamp does not amplify nor invert – the output tracks the input, albeit with an offset of about 0.3V. With a low-frequency sinusoidal input with amplitude 0.3V, the DC current was found to be 291μA with an AC ripple of 500nA amplitude. The fault-free DC current was 288μA with an 35μA AC component. Even in the presence of the fault, all the transistors continued to operate in the saturation region. This fault cannot be considered detectable if only the DC current were to be measured, but an RMS measurement of the AC component is probably sufficient.

When the input amplitude was increased to 2.0V, transistors M1 and M2 were forced into cutoff for part of the cycle, but the DC current remained almost unchanged at 292μA. Again this is not significantly different from the fault-free case. The AC component had an amplitude of 3μA. The apparently equivalent technique of varying the supply voltage had a much more significant effect. With the supply voltage, VDD at 3V and an input stimulus of 0.4V amplitude, again M1 and M2 operate for part of the time in cutoff, but now the DC current is 221μA, which is over 20% below the fault-free value and the fault must therefore be considered detectable.

It can therefore be seen that the basic hypothesis that causing transistors to switch between regions of operation is likely to increase the fault coverage is valid. This can be achieved by applying large input signals or by varying the supply voltage.

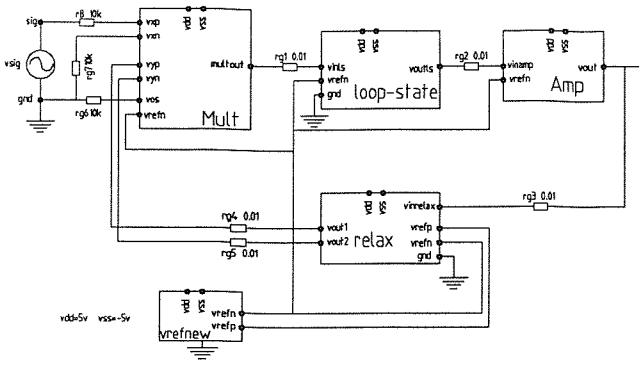


Figure 2 Phase-Locked Loop

In this example, varying the supply voltage had a much more significant effect than simply increasing the input signal amplitude.

The remainder of this paper is concerned with the application of this technique to a larger circuit.

Variable Supply Voltage Testing

A CMOS Phase-Locked Loop, Fig. 2, was used to test the application of the technique to a larger circuit. The circuit was designed using the MIETEC 2.4μm CMOS technology. The repeated insertion of circuit faults by hand is tedious, so the ANTICS fault simulator was used to inject faults into a SPICE netlist and to analyse the simulation results (5). Gate-source and gate-drain short fault models only were used. This is distinct from the open-gate fault model used in previously reported work. The total number of faults injected was 190, of which 28 were redundant (the short circuits already existed as part of the circuit configuration) and 33 were equivalent (the same inter-nodal short was injected at two separate transistors) Hence, 129 distinct faults were simulated. Monte Carlo simulations were performed for each of these 129 faults and the RMS value of the AC component of the supply current and the DC supply current were measured. The PLL was simulated as a whole, thus the input stimulus was a sinusoidal input within the locking frequency range.

The fault cover was then evaluated such that a fault was considered detectable if the 3σ points of the faulty and fault-free current distributions did not overlap. This separation or gap between two distributions is defined in (6) as:

$$gap = (\mu_f - 3\sigma_f) - (\mu + 3\sigma) \qquad (1)$$

where μ_f is the mean value of the faulty circuit response and μ is the mean value of the fault-free circuit response.

The fault covers were evaluated at different supply voltages using the DC and AC RMS values individually and combined.

A. VDD Change for whole PLL

Table I shows the fault cover for the PLL as the supply voltage was varied between 4.0V and 5.3V. As can be seen from Table 1, the fault cover changes very little for different supply voltages. The three figures given for each supply voltage value are the fault covers found by measuring the RMS value of the AC component of the supply voltage; the DC supply and the cumulative total of the two measures. A number of faults are detected by both measurements, which is why the sum is only around 10% higher than the RMS value alone.

Table II shows the cover obtained by combining two or more of the tests from Table I. All three given figures are fractionally above those of Table I. This means that a large common subset of faults is found by all the tests, with a small number of extra faults found by each test individually.

In other words, it appears that very few faults are sensitized, with respect to the supply current, by varying the supply voltage of the PLL as a whole.

B. Change in VDD of one subcircuit

Most of the undetected faults occured within the VCO, (relax in Fig. 2) therefore the exercise was repeated, with the supply voltage of the VCO subcircuit being changed to 4.5V while the supply voltage of the other subcircuits was held at 5.0V. The combined fault covers of this test and the previous tests are shown in Table III. The DC fault cover quoted here is that obtained using the first VDD value in the table. The cumulative totals for the single DC test and the two or three AC supply current tests are given.

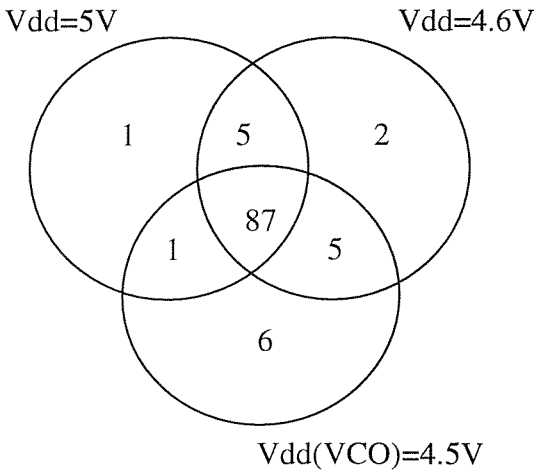


Figure 3 Venn diagram of fault covers of different tests.

Fig. 3 shows a Venn diagram for the fourth test in Table III. Again, it can be seen that most faults are detectable by all three tests. Six extra faults are detected by changing only the supply voltage of the VCO. While this only increases the overall fault cover by around 4%, these faults would not otherwise be detectable. Note that for all these tests the PLL continued to work as a PLL. (To be precise, for VDD=5V, and a 1MHz input sinusoid of 2V amplitude, the output of VCO is a square wave that varies between +1.96V and -2.3V, for VDD 4.5V and the same input the output value varies between +1.52V and -2.3V.)

C. VSS Change

Even by changing the supply voltage for just the VCO, a number of faults remained undetected. In addition, therefore VSS of the VCO was changed. Now, for VSS=-4.7V (instead of -5.0V) the fault coverage was 66% compared with a fault cover of 70% when VDD=4.7. Therefore monitoring the current from VSS seems to be better in terms of fault coverage.

For various VSS values of the VCO part (-4.6V, -4.65V, -4.7V, -4.75V, -4.8V) it was still not possible to detect any of those faults that couldn't be detected before.

TABLE I PLL FAULT COVER FOR VARYING SUPPLY VOLTAGE

VDD [V]	Fault Cover [%] RMS	Fault Cover [%] DC	Fault Cover [%] RMS + DC
5.3	67	38	74
5.0	64	40	73
4.8	67	39	77
4.7	69	39	77
4.6	68	44	77
4.5	67	40	77
4.0	65	41	76

TABLE II PLL FAULT COVER FOR COMBINED TESTS

Multiple VDDs [V]	Fault Coverage [%] RMS + DC
5 + 4.7 + 4.6	79
5.3 + 4.8 + 4.7 + 4.6	80
4.6 + 4.7	78

TABLE III CUMULATIVE FAULT COVER WITH VDD VARIED FOR ONE SUBCIRCUIT.

Multiple VDD [V]	Fault Cover [%] RMS + DC
5 + 4.7 + 4.6 + 4.5(VCO)	81
5.3 + 4.8 + 4.7 + 4.6 + 4.5(VCO)	82
4.6 + 4.7 + 4.5(VCO)	80
5 + 4.6 + 4.5(VCO)	83

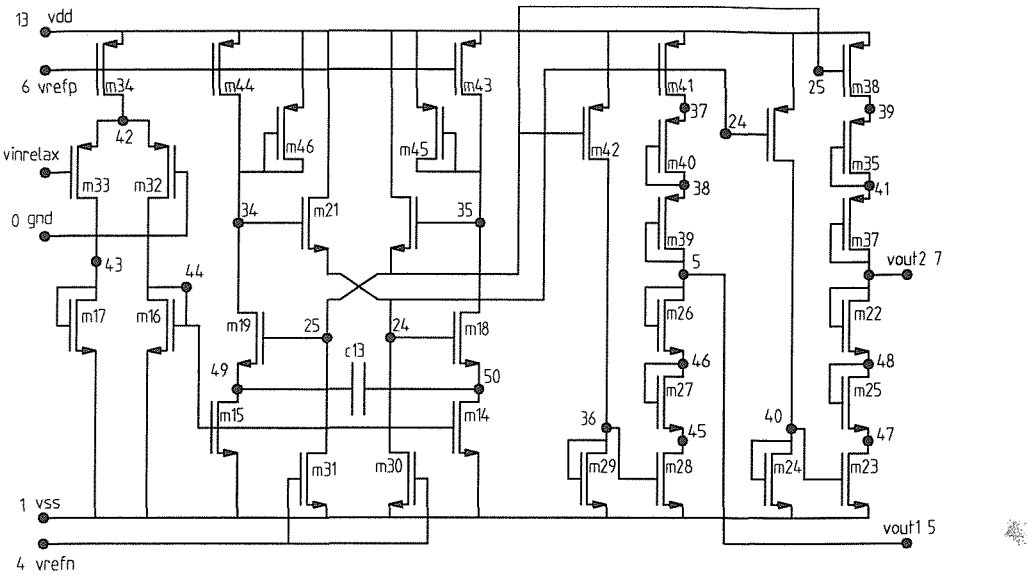


Figure 4 Voltage Controlled Oscillator

Undetected Faults

The fault cover using the DC and AC supply current tests was increased from 73% to 83% by using three different supply voltages. Nevertheless, 17% or 22 faults remain undetected. The reason why these faults remain undetected will be discussed here. 13 of the 22 undetected faults occur in the VCO. Fig. 4 shows the circuit diagram of the VCO. Of these 13 faults, 8 occur in the voltage divider chains on the right of the circuit in Fig. 4.These undetectable faults include the gate-source shorts on M26, M27 and M28. Although these faults would affect the functioning of the circuit, effect in terms of the supply current would be negligible, unless they were to force the PLL as a whole to cease to function. Therefore it is not surprising that a supply current test is unable to find them. More significantly, it should be noted that the original premise – that stimuli that would cause transistors to change their region of operation are ideal for supply current testing – cannot be satisfied for such circuit configurations. M26 and M27 are connected so as to be in permanent saturation.

Other undetectable faults include the gate to source shorts of M18 and M21 in Fig. 4. Again, from the circuit configuration, it would be almost impossible to apply any stimulus that would cause these transistors to switch their mode of operation. It is reasonable to suppose that this test technique cannot provide a significantly higher fault cover than that found here for voltage-mode circuits operating with ±5V supplies.

Conclusions

A testing technique for analog CMOS circuits has been discussed. This technique aims to sensitize faults by causing transistors to switch from their normal saturation mode of operation. Hence, the supply current, in both DC and AC domains is changed sufficiently to give a clear indication of the presence or absence of a fault. This technique has been shown to increase the fault cover of a complex analog CMOS circuit by 10% to 83%. This technique may be even more successful with lower voltage, current mode circuits. It is intended that the technique will form the basis of a mixed-signal Built-In Self-Test (BIST) methodology.

References

(1) E. Bruls, "Variable supply voltage testing for analogue CMOS and bipolar circuits.", International Test Conference, 1994.

(2) A.K.B. A'ain, A.H.Bratt and A.P. Dorey, "Testing analgoue circuits by power supply voltage control.", Electronics Letters, 3rd February 1994, Vol.30. No.3.

(3) A.K.B. A'ain, A.H.Bratt and A.P. Dorey, "Exposing Floating Gate Defects in Analogue CMOS Circuits by Power Supply Voltage Control Testing Technique.", 8th International Conference on VLSI Design, January 1995.

(4) A.K.B. A'ain, A.H.Bratt and A.P. Dorey, "Testing Analogue Circuits by AC Power Supply Voltage.", 9th International Conference on VLSI Design, January 1996.

(5) I. M. Bell, K. R. Eckersall, S. J. Spinks, G. E. Taylor, "Fault Oriented Test and Fault Simulation of Mixed Signal Integrated Circuits", Proc. of IEEE International Symposium on Circuits and Systems, Seattle, Washington, April 1995, pp.389-392

(6) Y. K. Malaiya, A. P. Jayasumana, Q. Tong, S. M. Menon, "Enhancement of Resolution In Supply Current Based Testing for large ICs", Bridging Faults and IDDQ Testing, IEEE Computer Society Press, Los Alamitos, California, 1992, pp 40-45.

(7) C. Chalk, M. Zwolinski "A Design for Test Technique to Increase the Resolution of Analogue Supply Current Tests", *ICECS'98*, Lisbon, 1998.

8.4.2 Y.Kılıç, C.D. Chalk and M. Zwolinski, "Design and Realisation of a new Built-In Current Sensor for Mixed-Signal IDDD Test", 5th IEEE International Mixed Signal Testing Workshop, Delta Whistler Resort, Vancouver (Whistler), British Columbia, Canada, 15-18 June 1999.

Design and Realisation of a new Built-In Current Sensor for Mixed-Signal I_{DDD} Test

Y. Kılıç , C.D. Chalk and M. Zwoliński

Department of Electronics and Computer Science
University of Southampton
Southampton SO17 1BJ, UK
{yk, mz}@ecs.soton.ac.uk

Abstract

A novel design of a built-in current sensor for analogue and mixed signal integrated circuits is proposed. The sensor is based on the principle of a shunt voltage regulator. The design is shown to be compact, with good performance.

Introduction

Supply current monitoring has proved to be an effective method for testing digital and analogue circuits [1-9]. It can be implemented either using automatic test equipment (ATE) or Built-In Current Sensors (BICS). Using a BICS is more advantageous in terms of: test equipment costs; increasing the testing rate; improving the fault detectability and observability of the Circuit Under Test (CUT); higher sensing current resolution and avoiding the influence of I/O currents which may dominate the chip's total current [2].

One way to increase the difference between the fault-free and faulty currents is to select a stimulus that causes the current flowing through the faulty components to dominate the supply current response. This technique has limitations, as it might not be possible to propagate the correct DC voltage or frequency of an AC stimulus to the CUT and there might be faults that remain undetected regardless of the stimulus.

A more direct approach is to partition the circuit into small blocks perhaps of similar complexity to an opamp and to measure the current from each with a BICS. In this paper we propose a new low cost BICS.

Built-In Current Sensors

Monitoring the supply current avoids the need to add intrusive circuitry that can load sensitive outputs or internal voltage nodes. All supply current monitoring techniques, however, suffer from poor resolution [10], particularly if measurements are taken off-chip. The situation is worse for IDDQ testing of digital ICs because the large capacitance between the supply terminal and ground and associated test equipment must be discharged before a static DC measurement can be taken. One possible solution is to add one or more BICS [11]. Sensors for digital applications cannot be used for monitoring analogue circuits since measurements are not taken continuously and they have extremely nonlinear transfer characteristics. Eckersall proposed using simple linear current mirrors monitoring each analogue macro within a two bit flash ADC [12].

In addition to the standard current mirror, Renovell [13] added voltage monitoring at the output and integrated the two measurements over time, thus producing a single measure of the functionality of the circuit. High fault coverages were quoted (98%) for the opamp tested.

There have been a number of BICS circuits proposed for digital applications [2-6, 8]. From our point of view, the most common drawback with most of those sensors is that they are not easily applicable to analogue and mixed-signal circuits since they require a large area for the serial active element. The most suitable sensor for analogue and mixed-signal applications is perhaps the one that was proposed in [6] and implemented in [7], Fig. 1. This sensor is based on a series voltage regulator. A series voltage regulator with a very small voltage drop is modified by including an extra transistor M2, which monitors the gate source voltage of the main series transistor M1. A change in the supply current drawn by the CUT will be seen as a change in the gate voltage of M1 as it tries to maintain the set voltage drop. The gates of M1 and M2 are connected together.

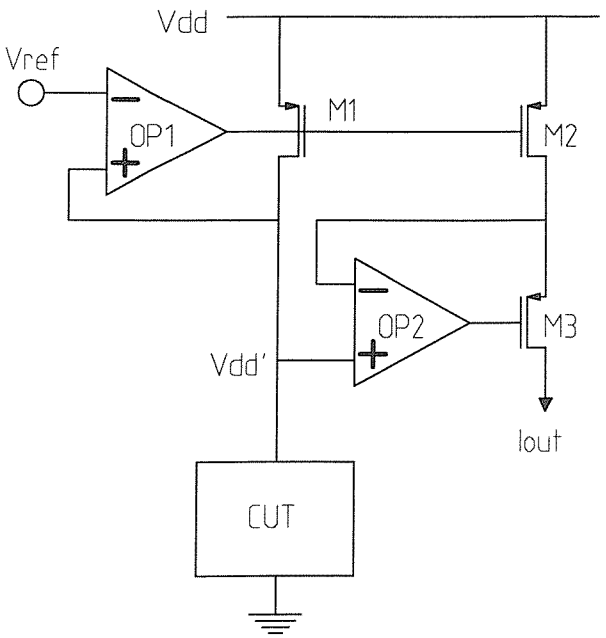


Figure 1 Series voltage regulator based BICS.

The current through this extra transistor M2 is therefore a copy of the supply current. OP2 and M3 are added to equalise the drain-source voltage of M1 and M2.

The main drawback of this sensor is the area required to realise transistor M1, which is almost as much as that of the two opamps. Since it is desirable to keep the voltage drop low (e.g. 50 mV for 5 V supply), the width/length ratio of M1 would be excessively large to sink the current through the CUT for analogue and mixed-signal circuits. In this paper we propose a new sensor to overcome this disadvantage. Our sensor is based on the shunt regulator principle.

Before explaining our new method let us first list the requirements for a current sensor.

Requirements of current sensors

The current sensor must produce an accurate copy of the current drawn by the CUT and not affect the performance of the circuit. It should fulfil the following requirements:

1. Frequency response comparable to the CUT
2. Ideal step response (no ringing or slew)
3. Low distortion
4. High signal to noise ratio (at least 20dB)
5. Low voltage drop in series with CUT (<50mV)
6. Zero input impedance
7. Tolerance to process parameter variation of sensor components.
8. Minimal area.

The New BICS Design

We use the shunt voltage regulator principle to derive a voltage proportional to the change in current of the CUT since we are interested only in the current variation of the CUT rather than the absolute DC power supply current. We can monitor the dynamic current flowing through the CUT by using the active shunt element. The area of the shunt element will depend on the current variation that the circuit experiences during normal operation. For many analogue circuits the power supply current variation can be less than one tenth of the quiescent bias current e.g. for a two stage CMOS opamp. Therefore, the size of the shunt transistor can be rather smaller than that of a series transistor.

Since the series element does not have to be an active device, it can easily be realised as a small value resistance, which will occupy a small area.

The shunt regulator based BICS is depicted in Fig. 2. The shunt element is formed by transistor M1, which is driven by the opamp. The opamp compares Vref to Vdiv, which is proportional to Vdrop, the voltage across Rdrop. The opamp and shunt transistor M1 will stabilise the voltage across Rdrop and so a constant current flows through Rdrop. If the load current changes by ΔIload there will be an equal but opposite change in the current through M1 of ΔIshunt. The transistors M1 and M2 share the same gate, source, and drain voltages. Therefore, the current through M2 will be proportional to Ishunt. The transistor M3 is included to compensate for any difference in the current through M1 and M2, which might occur due to channel length modulation. The current M2 is mirrored with M4 and M5 and applied to an external pin for further processing. This current can be converted directly to a voltage by connecting Iout to one terminal of a resistor with the other terminal connected to Vss. If a number of sensors are needed and their outputs need to be individually selected, it is best if Iout is not converted to a voltage. Poor quality switches can then be used to multiplex the outputs with no degradation in the signal fidelity. The output of the multiplexer can then be converted to a voltage and sent to an IC pin, if desired.

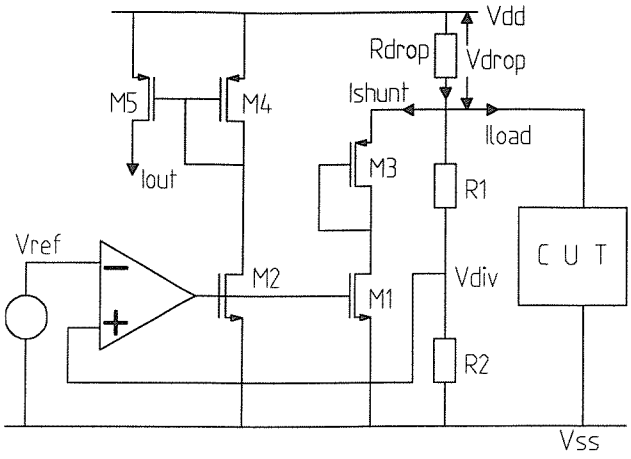


Figure 2 Shunt voltage regulator based BICS.

The value of Rdrop is calculated from (1),

$$R_{drop} = \frac{V_{drop}}{I_{load} + |\Delta I_{load}|} \quad (1)$$

and the width/length ratio of M1 is calculated from (2),

$$I_{shunt} = |\Delta I_{load}| = \frac{K_{sat}}{2}\frac{W}{L}(V_{gs} - V_t)^2(1 + \lambda V_{ds}). \quad (2)$$

where Ksat is the transconductance in saturation, Vgs = Vdd / 2. Let us assume that the example CUT draws 1mA DC supply current, 100μA AC supply current, and that Vdd is 5V. If the maximum value of Vdrop is 50mV, then the value of Rdrop is around 45Ω. To calculate the width/length ratio of M1, let us assume that the opamp has a maximum output voltage swing of 3V. If we assume that the typical value of Ksat is 17e-6 A/V^2 and the threshold voltage Vt is around 1V then the width-length ratio of the shunt transistor would be around 3. In contrast, if a BICS based on the series regulator principle were used, the width-length ratio of the series transistor would be 200. Also note that the ratio of the shunt transistor is independent of Vdrop and of the DC current drawn by the CUT. This is in contrast with the series regulator transistor, the width-length ratio of the series transistor is directly proportional to Vdrop and Ipsu.

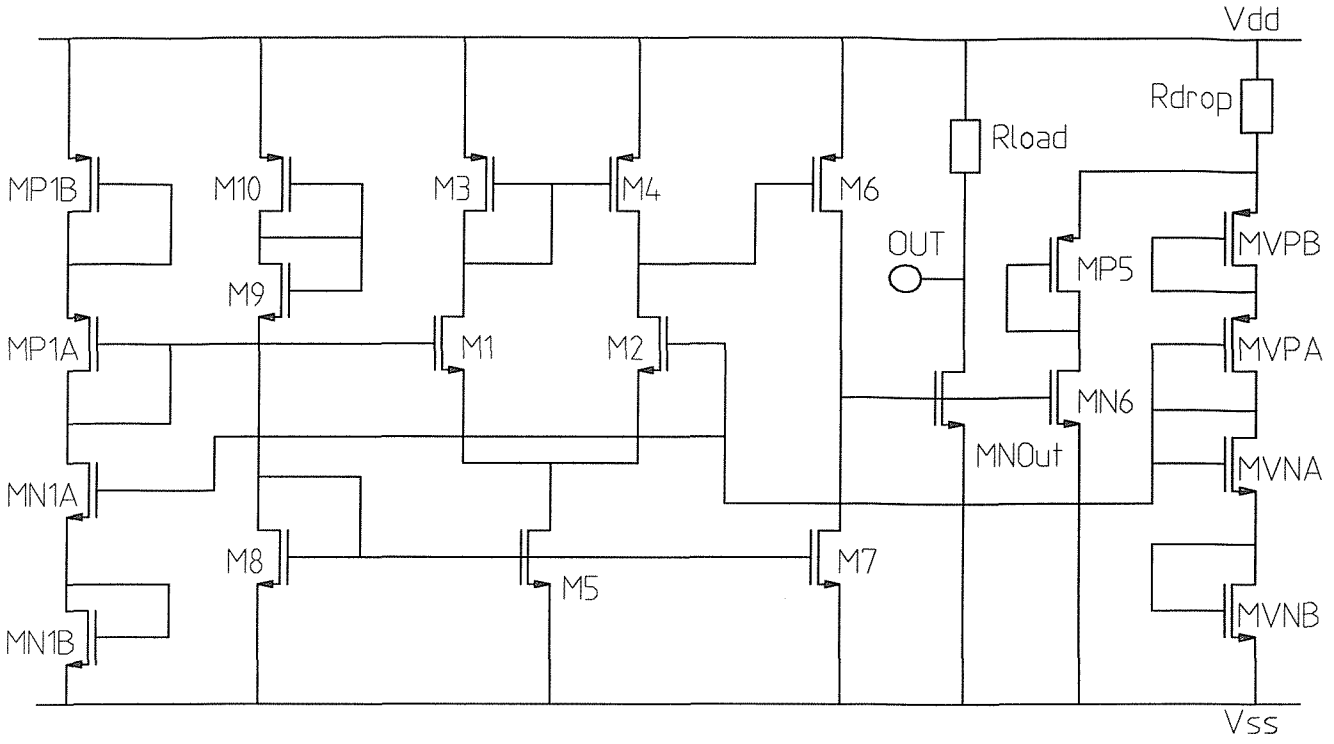


Figure 3 Miller opamp based CMOS implementation of BICS

Implementation of the BICS

Fig. 3 shows the practical CMOS implementation of the shunt regulator based BICS. The voltage divider is realised by a chain of MOSFETs: MVPB; MVPA; MVNA and MVNB. The voltage reference circuit is also implemented through a transistor chain: MP1B; MP1A; MN1A and MN1B, which is connected between Vdd and Vss.

Rdrop should have a small value in order to keep the voltage across it small. It is realised in poly resist. The main reason is that poly resist has a substantially smaller voltage dependence on the depletion-region width, compared to other types of resistance.

Since we want Rload to be relatively large, it has been realised as N-Well resist, which has a high resistance value per square. The comparator is realised as a two stage Miller opamp.

Simulation Results

An HSPICE analysis of the netlist of Fig. 3 showed that the BICS has a 6 MHz frequency break point. The frequency response is shown in Fig. 4.

After laying out the circuit, post simulation results shows that the new BICS is working as expected. Post simulations of the netlist extracted from the BICS layout showed that the new frequency bandwidth of the BICS is 2.4 MHz. The frequency analysis of the netlist extracted from the layout is given in Fig. 5. This decrease in the frequency response is due to the parasitic capacitance values of interconnects and pads. The circuit has been fabricated in 2.4μ CMOS MIETEC technology [14]. The layout of the fabricated chip is given in Fig 6.

We have carried out a number of measurements on the fabricated circuit to confirm the simulation results. The voltage drop across the shunt resistor Rdrop is found to be around 46 mV, which is very near to the expected Vdrop value of 50 mV. The measurements showed that the BICS frequency bandwidth is around 2 MHz.

Conclusions

In this paper we have described a new Built-In Current Sensor based on a shunt regulator principle. Unlike the sensor proposed in [7] which is based on the series regulator principle, the area occupied by the sensor is dependent on the magnitude of the current being measured. Since the AC current drawn by typical analogue circuits is less than the DC current, the area overhead should be much lower than that for a sensor based on the series regulator. The sensor was found to be less sensitive to process parameter variations than the circuit under test. The sensor also has a frequency response comparable to the circuit under test and has a Vdrop of less than 50 mV.

Acknowledgements

This work has been partly supported by EPSRC. The authors would also like to thank the following MSc and MEng students for their contributions to this work: Bruno Johnson, Tom Edwards and Aysel Yildiz.

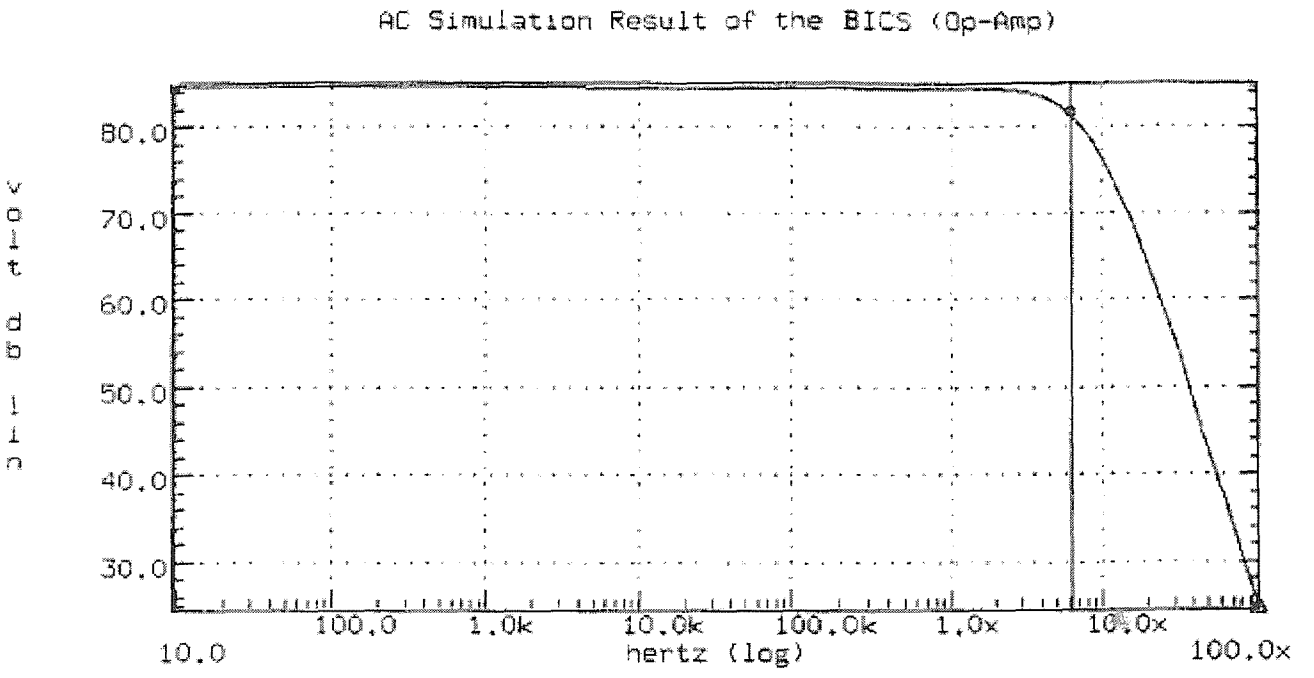


Figure 4. Frequency analysis of the BICS before layout.

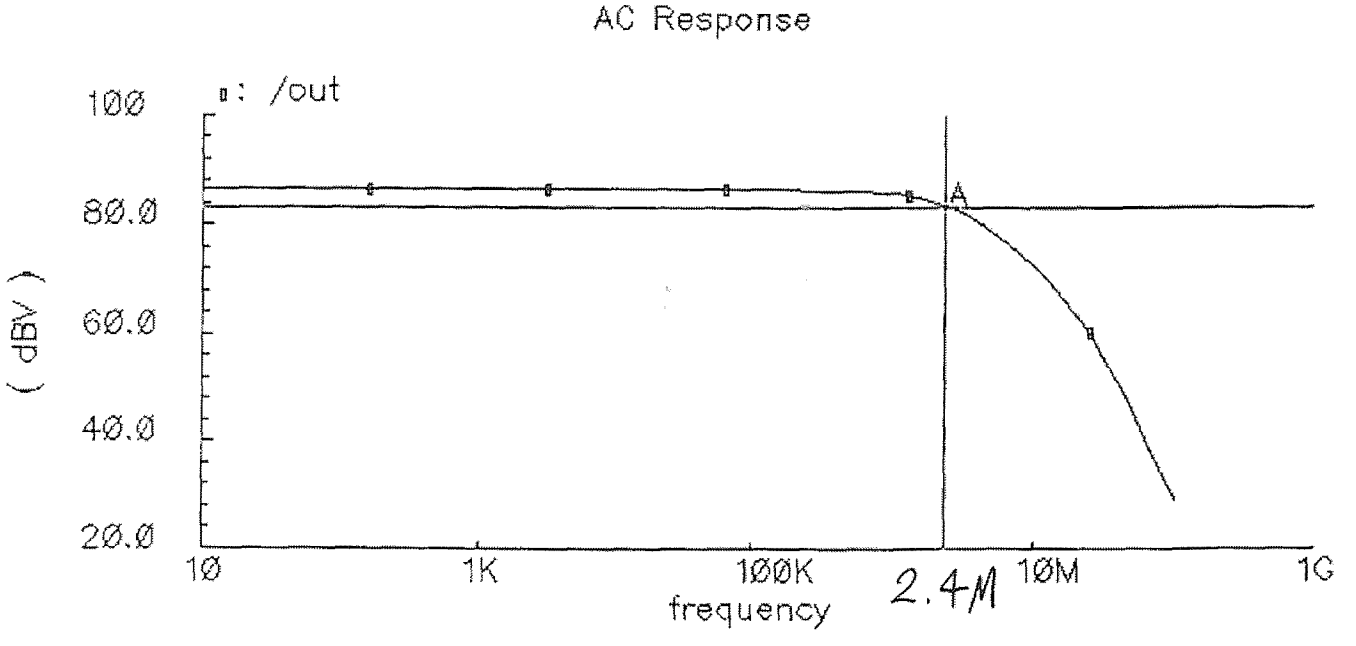


Figure 5. Frequency analysis of the BICS after the layout.

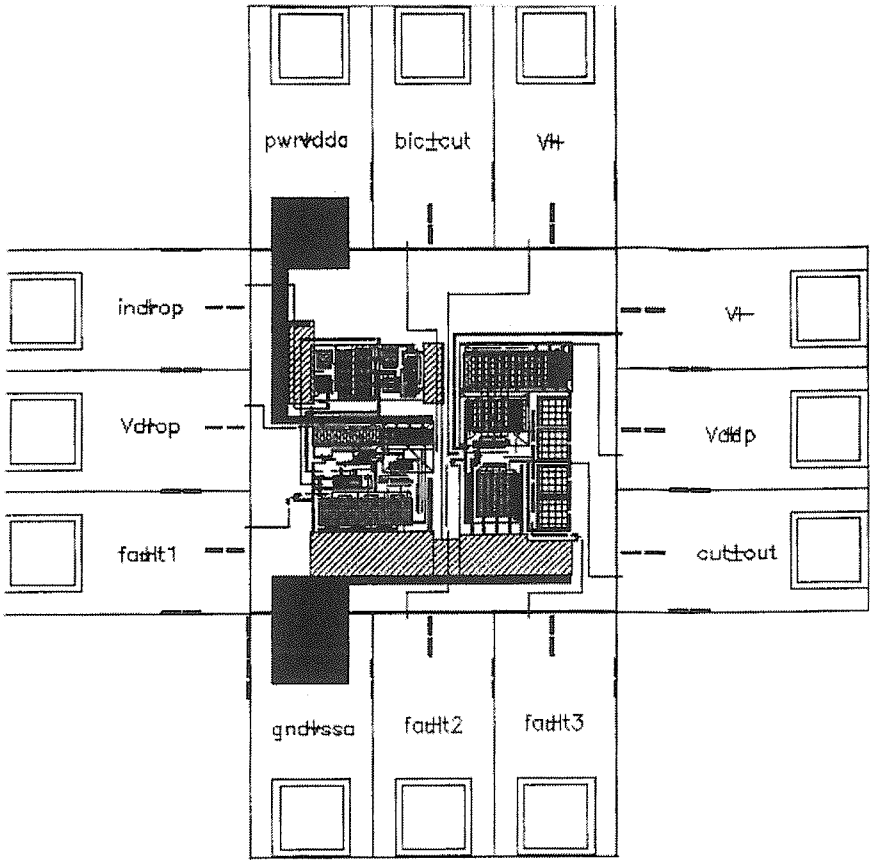


Figure 6. Layout view of the designed BICS with CUT.

References

[1] Y. Kılıç and M. Zwoliński, " Testing Analog Circuits by Supply Voltage Variation and Supply Current Monitoring", The 1999 IEEE Custom Integrated Circuits Conference, May 16-19, 1999, Town and Country Hotel, San Diego, California, USA.

[2] Jing-Jou Tang, Kuen-Jong Lee, and Bin-Da Liu, "A Practical Current Sensing Technique for I_{DDQ} Testing", IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Vol.3, No.2, June 1995, pp. 302-310.

[3] J. Segura, M. Roca, D. Mateo and A. Rubio, "Built-in dynamic current sensor circuit for digital VLSI CMOS testing", Electronics Letters, 29th September 1994, Vol.30, No.20, pp. 1668-1669.

[4] Kuen-Jong Lee, Kou-Shoung Huang and Min-Cheng Huang, "Low voltage built-in current sensor", Electronics Letters, 10th October 1996, Vol.32, No.21, pp. 1942-1943.

[5] Kuen-Jong Lee and Jing-Jou Tang, "A Built-In Current Sensor Based on Current-Mode Design", IEEE Transactions on Circuits and Systems-II: Analog and Digital Signal Processing, Vol.45, No.1, January 1998, pp. 133-137.

[6] Jeong Beom Kim, Sung Je Hong, and Jong Kim, "Design of a Built-In Current Sensor for I_{DDQ} Testing", IEEE Journal of Solid-State Circuits, Vol.33, No.8, August 1998, pp.1266-1272.

[7] Seevinck M., "Analog Interface Circuits for VLSI", in Analogue IC Design: the current mode approach, 1990, Toumazou C., Lidgey F. J., and Haigh D. G. (eds.) Peter Peregrinus Ltd. IEE, London, pp. 451-489.

[8] Karim Arabi and Bozena Kaminska, "Design and Realization of an Accurate Built-In Current Sensor for On-Line Power Dissipation Measurement and I_{DDQ} Testing", International Test Conference, 1997, pp.578-586.

[9] Zwolinski M, Chalk C., Wilkins B.R. and Suparjo B.S., "Analogue Circuit Test using RMS Supply Current Monitoring", 2nd IEEE International Mixed Signal Testing Workshop, 1996.

[10] Richardson A., Bratt A,. Baturone I., and Huertas J.L. :"The Application of I_{DDX} Test Strategies in Analog and Mixed Signal Ics", IEEE Int. Mixed Signal Testing Workshop, 1995, Grenoble, France.

[11] Hawkins C.F., Soden J.M., Fritzmeier R.R., and Horning L.K.: "Quiescent Power Supply Current Measurement for CMOS IC Defect Detection", IEEE Transactions on Industrial Electronics, 1989, Vol.36, No.2, pp.211-218.

[12] Eckersall K.R., Wrighton P.L., Bell I.M., Bannister B.R., and Taylor G.E., "Testing Mixed Signal ASICs through the use of Supply Current Testing", Proc. 3rd European Test Conf., Rotterdam, The Netherlands, April 1993, pp.385-391.

[13] Renovell M., Azais F., and Bertrand Y., "On-chip signature analyser for analogue circuit testing", Electronics Letters, 1996, Vol.32, No.24, pp. 2185-2186.

[14] A. Yildiz, "Implementation of the Built-in Current Sensor for Analogue Circuits", an MSc dissertation, Dept. of ECS, University of Southampton, October 1999.

8.4.3 Y. Kilic, M. Zwolinski, "Concurrent Transient Fault Simulation of Nonlinear Analogue Circuits", 5th IEEE International Mixed Signal Testing Workshop, Montpellier, France, June 21-23, 2000.

Concurrent Transient Fault Simulation of Nonlinear Analogue Circuits

Yavuz Kılıç and Mark Zwoliński

Department of Electronics and Computer Science
University of Southampton
Southampton SO17 1BJ, UK
{yk, mz}@ecs.soton.ac.uk

Abstract

In this paper, a new approach for transient concurrent analogue fault simulation is presented. The effectiveness of the technique is evaluated through the use of IEEE Mixed-signal benchmark circuits.

1 Introduction

Fault diagnosis techniques need fault models. Simulation-before-test techniques are best suited for detecting catastrophic faults and local parametric faults. They perform less well in detecting global parametric faults, since for such faults the separation between faulty and fault-free responses is less marked. Simulation-after-test techniques, however, are better suited for detecting problems with global parametric variations and mismatch, but are not well-suited for detecting catastrophic faults [1].

Analogue fault simulation is the first step to fault coverage analysis, fault grading, fault collapsing, and BIST [2]. Fast fault simulation of analogue circuits is crucially important in terms of speeding up the analogue testing process. As can be seen from the Fig.1, any speed up in test and debug time after first silicon directly affects a reduction in time-to-market [1].

There are still not many analogue fault simulation algorithms in contrast to digital fault simulation algorithms. This is partly because there is not yet a standard fault model, such as the stuck-at fault model in digital circuits, for analogue circuits. Without a fault model, concepts such as fault cover and fault detectability are difficult to quantify. Therefore the design of fault simulation algorithms that exploit such information is difficult.

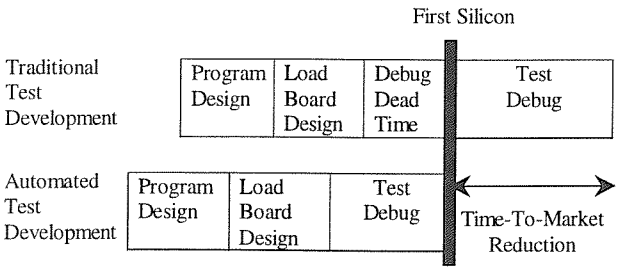


Figure 1. Ideal potential time-to-market reduction due to the use of electronic design automation (EDA) tools for test development [1].

Nevertheless, it is generally acknowledged that analogue fault simulation is slow. To make analogue fault simulation usable, new techniques need to be developed to cut the CPU time. Some approaches to speeding up the analogue fault simulation process are: dynamic fault dropping and/or collapsing (in which defects that cause similar change in the faulty circuit response compared with the fault-free one and/or compared with another faulty circuit response are considered equivalent); behavioural modelling (whereby parts of the circuit are modelled at a more abstract level, therefore reducing simulation complexity and time) and lastly new algorithmic techniques.

There has been some research on new algorithms for speeding up analogue fault simulation process [3], [4], [5]. In [4], a concurrent fault simulator (CONCERT) for nonlinear analogue circuits is presented. In order to speed up the simulation process, the authors use fault ordering, state prediction and reduced-order fault matrix computation.

In [5], the authors consider only linear mixed-mode circuits. They attempted to represent the analogue blocks in the sampled Z-domain using discrete-time state equations in order to provide a common framework for the simulation of both analogue and digital blocks. Their fault simulator consists of two modules: The first module performs serial fault simulation of the analogue block, and the second performs parallel or concurrent fault simulation of the digital block. The fault detection is done at the digital outputs.

The objective of this work is to speed up analogue fault simulation. As stated above, no matter which analogue fault simulation method is used, dynamic fault dropping and/or fault collapsing is one way to speed up the analogue fault simulation process. To achieve this goal, we need to compare the faulty circuit response with the fault-free one and/or with other faulty versions of the circuit responses. If the compared circuit responses are within a specified threshold then we can either drop or collapse the fault, removing the faulty circuit from further analysis. Hence, an efficient closeness measurement is required to carry out this comparison.

2 Closeness Measurement

There are number of closeness measurements available in the literature but they are not easily applicable to the

problem of analogue fault simulation [7]. In the analogue domain, the most widely used closeness measurement has been the Euclidean distance [8], [9].

In [8], the normalised absolute distance is used. The single point closeness measurement is adopted in order to detect the similarity between faulty responses and the fault-free response in DC fault simulation. The distance between each fault response and the fault-free response (at the end of each N-R (Newton-Raphson) iteration) is measured. The faults are then ordered such that the first fault to be simulated has the minimum distance to the fault-free circuit response. This ordering process is repeated for all the faults. There is no discussion however of when to stop this ordering process and how to decide which fault is detectable.

In [9], it is suggested that taking a single point closeness measurement into account might not accurately detect the distance between the fault-free circuit response and faulty circuit responses, as the distance might not always be the same during the simulation. Therefore a multi point closeness measurement for the DC fault simulation of analogue circuits is used.

In the multi-point closeness measurement, let x_f be the vector containing the node voltages and/or branch currents of the faulty circuit and x_g be the vector of the same elements for the fault-free circuit. The distance between the vectors x_f and x_g is given as follows [9]:

$$d_{fg} = \frac{1}{M}\sum_{m=1}^{M}\left\| x_g^m - x_f^m \right\| \qquad (2\text{-}1)$$

where d_{fg} is the distance between the vectors x_f and x_g. x_g^m and x_f^m are the responses for the good circuit and the f^{th} faulty circuit at the m^{th} N-R iteration and M is the number of consecutive N-R iterations for the closeness measurement.

This closeness measurement is used to decide whether the faulty circuit response is close enough to the fault-free circuit response within a specified threshold to drop this fault from the fault list at the succeeding iteration step. If a faulty response is far enough from the fault-free response within a specified threshold we then can drop this fault also from the fault list, as this fault is deemed detectable. We use the same closeness measure with a different threshold to determine this distance between faulty and fault-free responses.

Since we do the closeness measurement for transient fault simulation of nonlinear analogue circuits, (in contrast to the work reported in [9]), instead of using a number of N-R iterations for the multi-point closeness check, we use a number of time points. Assuming that x_f is the vector containing the node voltages and/or branch currents of the faulty circuit and x_g is the vector of the same elements for the fault-free circuit, the distance between the vectors x_f and x_g is now given as:

$$d_{fg} = \frac{1}{n}\sum_{i=1}^{n}\left\| x_g^i - x_f^i \right\| \qquad (2\text{-}2)$$

where d_{fg} is the distance between the vectors x_f and x_g. x_g^i and x_f^i are the responses for the good circuit and the k^{th} faulty circuit at the i^{th} time point and n is the number of consecutive time points for the closeness measurement.

In this work, we employ both normalised absolute and relative Euclidean norms in order to do the closeness check. We also allow a choice between those two distance measures with different options: single point single node, single point multi-node, multi-point single node, multi-point multi-node.

2.1 Threshold Calculation

Let a_f be the node voltage and/or branch current of a faulty circuit and a_g be the same value for the fault-free circuit in the case of the single node closeness check. If we assume that the closeness measure is the relative Euclidean norm and $a_f = ka_g$, then the threshold is given as:

$$th = \left\| a_g - a_f \right\| = \left\| a_g - ka_g \right\| = \left| 1-k \right| a_g \qquad (2\text{-}3)$$

where th is the threshold between the faulty value and the fault-free value.

If the closeness measure is the normalised absolute Euclidean norm then the threshold is:

$$th = \left| \frac{a_g - a_f}{a_g} \right| = \left| \frac{a_g - ka_g}{a_g} \right| = \left| 1-k \right| \qquad (2\text{-}4)$$

In the case of the multi-node closeness check, each element of the faulty vector might not be perturbed by the same amount (k) as a result of the fault being injected in the fault-free circuit. There will be a vector k consisting of elements (k_1, k_2, ... , k_l) where l is number of nodes in the circuit being simulated. In order to simplify the calculations, however, we assume that $k_1=k_2=...=k_l=k$.

Now we can say that we want to drop the fault when the faulty response is within the 5% of the fault-free response (thus k is 0.95) or 35% far from the fault-free response (where k is 0.65).

3 Concurrent Fault Simulation

Concurrent fault simulation is well understood for digital circuits [6]. Recently, the principle of concurrent fault simulation has been applied to analogue circuits [3], [4]. In concurrent transient analogue fault simulation, all the faulty circuits along with the fault-free circuit are simulated simultaneously at a time point before simulation proceeds to the next time step. Different time steps may be used to simulate each faulty version of the circuit, in which case the simulation will speed up since some faulty versions might take fewer time steps to simulate. This is distinct from the work done in [4] and [9]. Further, if the terminal value of the faulty device is close enough to the terminal value of the fault-free device within a specified threshold then the fault-free device values can be reused for the faulty one, thus reducing the CPU time.

3.1 Concurrent Fault Simulation Algorithm

In this work, we have used structural short circuit faults, which are constructed from the fault-free circuit netlist. If the fault free circuit has n nodes then we have $n(n\text{-}1)/2$ faulty versions of the circuit to analyse as we assume that there is a short circuit fault between each pair of nodes in the fault free circuit. We assign a resistance value of 10 ohms to each short circuit fault.

The steps for the algorithm that has been implemented in our own analogue circuit simulator are as follows:

Step 1. Constitute the original fault list, $\beta^0 = \{F^1, F^2, \ldots, F^N\}$, by inserting all possible short faults ($N=n(n\text{-}1)/2$) into the fault-free circuit to obtain N faulty versions. Here β^0 represents the original fault list and F^k is the kth fault. Let x_0^i, $i \in [0, \infty)$ be the response vector of the fault-free circuit at the ith time point and x_f^i, $i \in [0, \infty)$ be the response vector of the fth faulty circuit at the ith time point during transient concurrent fault simulation.

Step 2. Simulate all faulty circuits along with the fault-free for a user-defined number of time points. Now, x_0^i and x_f^i are available.

Step 3. Carry out a user-defined closeness measurement between each faulty circuit response, x_f^i, and the fault-free circuit response x_0^i. If a faulty circuit response is either within or far enough outside user-defined thresholds then drop this fault from the original fault list.

Dropped faults are not simulated further during the transient analysis. In other words it is implicitly assumed that for a given test stimulus, faults that cause sufficiently different behaviour from the fault-free are detectable and do not require further analysis, and that faults that are so far indistinguishable from the fault-free behaviour are also not worthy of further consideration. The interval over which the closeness check is performed may be chosen by the user. Clearly, it makes sense to perform the closeness check only once steady state has been reached (for a periodic stimulus). Similarly, performing a closeness check and then applying a significantly different stimulus would not be sensible as the dropped faults would not be tested against that new stimulus. Non-convergent faults are automatically dropped.

4 Examples

We used an inverter circuit consisting of two MOS transistors and a capacitance, Fig. 2, and a 2-stage Miller opamp (from the IEEE Mixed-signal benchmark circuits suite) consisting of 9 MOS transistors, a resistance and a capacitance, Fig. 3, in order to validate our algorithm.

For the inverter circuit we used a pulse signal as the input stimulus, the single point single node Euclidean norm as the distance norm, 5% and 35% margins for closeness and "farness". The transient analysis was run over 9 periods of the input stimulus and the distance check was carried on the second period of the output signal. There are 6 possible short faults. If we don't use fault dropping then the total number of device evaluations for during the concurrent fault simulation of these 6 faults is 23880 and the CPU time (which is obtained by doing the simulations five time and taking the average) was 4208ms. With fault dropping the total number of device evaluations was 9054, which represents 62% saving, and the CPU time was 1536ms, which is a 64% increase in the speed of the simulation. 3 faulty responses are very close to and other 3 are very far from the fault-free response so that all 6 faults are dropped from further simulation.

For the 2-stage opamp we use a sine wave as the input stimulus, and again we used the single point single node Euclidean norm where looking at the output node for the distance check. We had 56 possible short faults, but 6 of them failed DC convergence. We therefore simulated in total 50 short faults. 11 faulty responses were very close to and 38 faulty responses were very far from the fault-free response. The speed-up was 65% in terms of total number of device evaluations and 63% in terms of the CPU time.

5 Conclusions and Future Work

We have shown that using our new algorithm is effective in speeding up analogue fault simulation. We have not considered fault collapsing in this work. We will look at fault collapsing to even more increase the speed of the simulation of non-linear analogue circuits.

6 References

[1] Linda S. Milor, "A Tutorial Introduction to Research on Analog and Mixed-Signal Circuit Testing", IEEE Transactions on Circuits and Systems-II, Analog and Digital Signal Processing, Vol. 45, No. 10, October 1998.

[2] B.Vinnakota, "Analog and Mixed-Signal Test", Prentice Hall PTR, 1998.

[3] M. Zwolinski, A. D. Brown, C. D. Chalk, "Concurrent Analogue Fault Simulation", 3rd IMSTW, June 3-6, 1997, Seattle, Washington USA.

[4] J. Hou and A. Chatterjee, "CONCERT: A Concurrent Transient Fault Simulator for Nonlinear Analog Circuits", IEEE/ACM Int. Conf on CAD, Digest of Technical Papers, p 384-491, Nov 8-12, 1998.

[5] A. Balivada, et. al., "A Unified Approach for Fault Simulation of Linear Mixed-Signal Circuits", Journal of Electronic Testing, Theory and Applications 9, 29-41 (1996), 1996 Kluwer Academic Publishers, manufactured in The Netherlands.

[6] Abramovici, M., Breuer, M.A., Friedman, A.D, "Digital Systems Testing and Testable Design", IEEE Press, 1990.

[7] Chi-hau Chen, *Statistical Pattern Recognition*, Hayden Book Company, Inc., Rochelle Park, New Jersey, 1973.

[8] Michael W. Tian, C.-J. Richard Shi, "Efficient DC Fault Simulation of Nonlinear Analog Circuits",

Design, Automation and Test in Europe Conference (DATE98), Paris, France, Feb. 23-26, 1998.

[9] Z. R. Yang and M. Zwolinski, "Fast, Robust DC and Transient Fault Simulation of Nonlinear Analogue Circuits", Design, Automation and Test in Europe Conference (DATE99), Munich, Germany, March 9-12, 1999.

[10] http://www.ee.washington.edu/mad/benchmarks/benchmarks.html

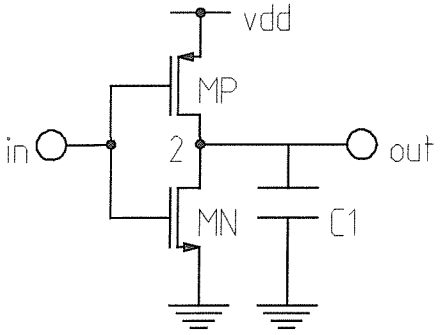


Figure 2. CMOS Inverter

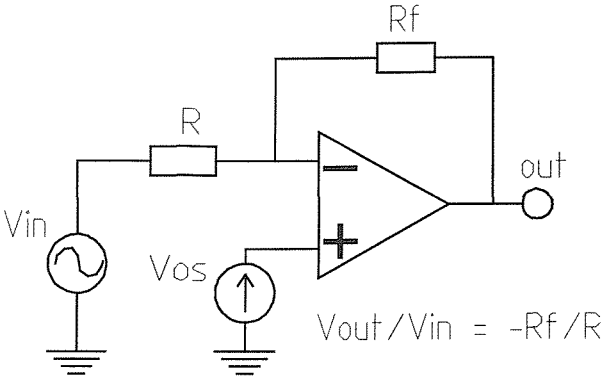


Figure 3. Inverting amplifier using opamp of Figure 4.

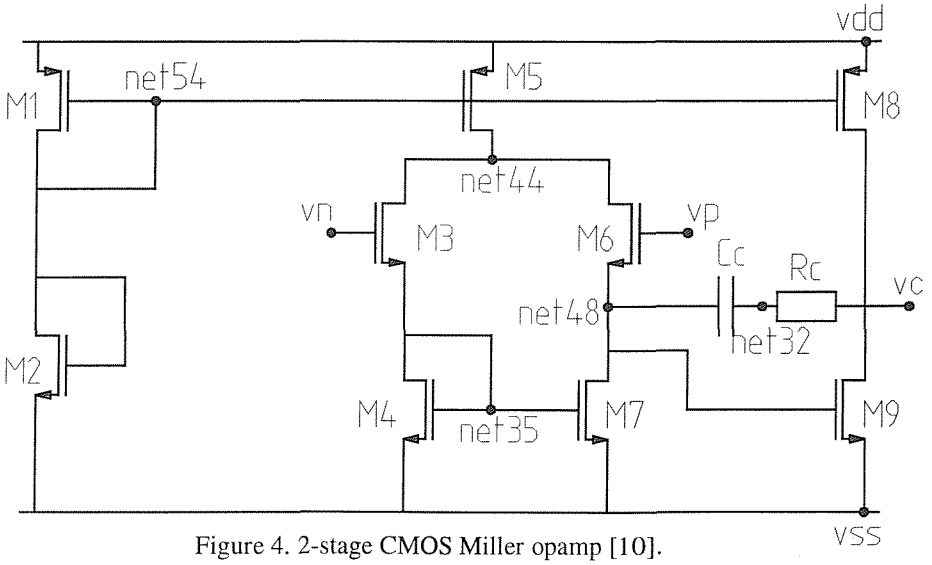


Figure 4. 2-stage CMOS Miller opamp [10].

8.4.4 Mark Zwoliński and Yavuz Kılıç, "Closeness Measurement in Concurrent Analogue Fault Simulation", ICSES '2000 International Conference on Signals and Electronic Systems, 17 - 20 October 2000 USTROŃ, POLAND.

Closeness Measurement in Concurrent Analogue Fault Simulation

Mark Zwoliński and Yavuz Kılıç

Department of Electronics and Computer Science, University of Southampton, Southampton SO17 1BJ, UK, email mz@ecs.soton.ac.uk

ABSTRACT

A new approach for transient concurrent analogue fault simulation is presented. Metrics for measuring the closeness of simulation responses are discussed. The effectiveness of the technique is evaluated through the use of IEEE Mixed-signal benchmark circuits.

Keywords: Analogue fault simulation, Analogue fault modelling, Transient analysis

1. INTRODUCTION

Fault diagnosis techniques need fault models. Simulation-before-test techniques are best suited for detecting catastrophic faults and local parametric faults. They perform less well in detecting global parametric faults, since for such faults the separation between faulty and fault-free responses is less marked. Simulation-after-test techniques, however, are better suited for detecting problems with global parametric variations and mismatch, but are not well-suited for detecting catastrophic faults [1].

Analogue fault simulation is the first step in fault coverage analysis, fault grading, fault collapsing, and BIST [2]. Fast fault simulation of analogue circuits is crucially important in terms of speeding up the analogue testing process. Any speed up in test and debug time after first silicon directly affects a reduction in time-to-market [1].

There are still not many analogue fault simulation algorithms, in contrast to digital fault simulation techniques. This is partly because there is not yet a standard fault model, such as the stuck-at fault model in digital circuits, for analogue circuits. Without a fault model, concepts such as fault cover and fault detectability are difficult to quantify. Therefore the design of fault simulation algorithms that exploit such information is difficult.

Nevertheless, it is generally acknowledged that analogue fault simulation is slow. To make analogue fault simulation usable, new techniques need to be developed to cut the CPU time. Some approaches to speeding up the analogue fault simulation process are: dynamic fault dropping and/or collapsing (in which defects that cause similar change in the faulty circuit response compared with the fault-free one and/or compared with another

This work has been supported by EPSRC grant GR/L35829.

faulty circuit response are considered equivalent); behavioural modelling (whereby parts of the circuit are modelled at a more abstract level, therefore reducing simulation complexity and time) and lastly new algorithmic techniques.

There has been some research on new algorithms for speeding up analogue fault simulation process [3], [4], [5]. In [4], a concurrent fault simulator (CONCERT) for nonlinear analogue circuits is presented. To speed up the simulation process, fault ordering, state prediction and reduced-order fault matrix computation are used.

In [5], only linear mixed-mode circuits are considered. Analogue blocks are represented in the sampled Z-domain using discrete-time state equations to provide a common framework for the simulation of both analogue and digital blocks. The fault simulator consists of two modules: The first module performs serial fault simulation of the analogue block, and the second performs concurrent fault simulation of the digital block. Fault detection is done at the digital outputs.

The objective of this work is to speed up analogue fault simulation. As stated above, no matter which analogue fault simulation method is used, dynamic fault dropping and/or fault collapsing can speed up the analogue fault simulation process. To achieve this goal, we need to compare the faulty circuit response with the fault-free and/or with other faulty versions of the circuit responses. If the compared circuit responses are within a specified threshold we can either drop or collapse the fault, removing the faulty circuit from further analysis. Hence, an efficient closeness measurement is required to carry out this comparison.

2. CLOSENESS MEASUREMENT

There are number of closeness measurements available in the literature but they are not easily applicable to the problem of analogue fault simulation [7]. In the analogue domain, the most widely used closeness measurement has been the Euclidean distance [8], [9].

In [8], the normalised absolute distance is used. The single point closeness measurement is adopted in order to detect the similarity between faulty responses and the fault-free response in DC fault simulation. The distance between each fault response and the fault-free response (at the end of each Newton-Raphson (NR) iteration) is measured. The faults are then ordered such that the first fault to be simulated has the minimum distance to the fault-free circuit response. This ordering process is repeated for all the faults. There is no discussion however of when to stop this ordering process and how to decide which fault is detectable.

In [9], it is suggested that taking a single point closeness measurement into account might not accurately detect the distance between the fault-free circuit response and faulty circuit responses, as the distance might not always be the same during the simulation. Therefore a multi point closeness measurement for the DC fault simulation of analogue circuits is used.

In the multi-point closeness measurement, let x_f be the vector containing the node voltages and/or branch currents of the faulty circuit and x_g be the vector of the same elements for the fault-free circuit. The distance between the vectors x_f and x_g is given as follows [9]:

$$d_{fg} = \frac{1}{M}\sum_{m=1}^{M}\left\|x_g^m - x_f^m\right\| \tag{1}$$

where d_{fg} is the distance between the vectors x_f and x_g. x_g^m and x_f^m are the responses for the good circuit and the f^{th} faulty circuit at the m^{th} NR iteration and M is the number of consecutive NR iterations for the closeness measurement.

This closeness measurement is used to decide whether the faulty circuit response is close enough to the fault-free circuit response to drop this fault from the fault list at the succeeding iteration step. If a faulty response is far enough from the fault-free response we can similarly drop this fault also from the fault list, as this fault is deemed detectable. We use the same closeness measure with a different threshold to determine the distance between faulty and fault-free responses.

Since we do the closeness measurement for transient fault simulation of nonlinear analogue circuits, (in contrast to the work reported in [9]), instead of using a number of NR iterations for the multi-point closeness check, we use a number of time points. Assuming that x_f is the vector containing the node voltages and/or branch currents of the faulty circuit and x_g is the vector of the same elements for the fault-free circuit, the distance between the vectors x_f and x_g is now given as:

$$d_{fg} = \frac{1}{n}\sum_{i=1}^{n}\left\|x_g^i - x_f^i\right\| \tag{2}$$

where d_{fg} is the distance between the vectors x_f and x_g. x_g^i and x_f^i are the responses for the good circuit and the k^{th} faulty circuit at the i^{th} time point and n is the number of consecutive time points for the closeness measurement.

In this work, we employ both absolute and relative normalised Euclidean norms in order to do the closeness check. We also allow a choice between those two distance measures with different options: single point single node, single point multi-node, multi-point single node, multi-point multi-node.

2.1. Threshold Calculation

Let a_f be the node voltage and/or branch current of a faulty circuit and a_g be the same value for the fault-free circuit in the case of the single node closeness check. If we assume that the closeness measure is the relative Euclidean norm and $a_f = ka_g$, then the threshold is:

$$th = \left\|a_g - a_f\right\| = \left\|a_g - ka_g\right\| = \left|1-k\right|a_g \tag{3}$$

where th is the threshold between the faulty value and the fault-free value.

For the normalised absolute Euclidean norm the threshold is:

$$th = \left|\frac{a_g - a_f}{a_g}\right| = \left|\frac{a_g - ka_g}{a_g}\right| = \left|1-k\right| \tag{4}$$

In the case of the multi-node closeness check, each element of the faulty vector might not be perturbed by the same amount (k) as a result of the fault being injected in the fault-

free circuit. There will be a vector k consisting of elements (k_1, k_2, ... , k_l) where l is number of nodes in the circuit being simulated. In order to simplify the calculations, however, we assume that $k_1=k_2=...=k_l=k$.

Now we can say that we want to drop the fault when the faulty response is within the 5% of the fault-free response (k is 0.95) or 35% far from the fault-free response (k is 0.65).

3. CONCURRENT FAULT SIMULATION

Concurrent fault simulation is well understood for digital circuits [5]. Recently, the principle of concurrent fault simulation has been applied to analogue circuits [3], [4]. In concurrent transient analogue fault simulation, all the faulty circuits along with the fault-free circuit are simulated simultaneously at a time point before simulation proceeds to the next time step. Different time steps may be used to simulate each faulty version of the circuit, in which case the simulation will speed up since some faulty versions might take fewer time steps to simulate. This is distinct from the work done in [4] and [9]. Further, if the terminal value of the faulty device is close enough to the terminal value of the fault-free device within a specified threshold then the fault-free device values can be reused for the faulty one, thus reducing the CPU time.

3.1. Concurrent Fault Simulation Algorithm

In this work, we have used structural short circuit faults, which are constructed from the fault-free circuit netlist. If the fault free circuit has n nodes then we have $n(n\text{-}1)/2$ faulty versions of the circuit to analyse as we assume that there is a short circuit fault between each pair of nodes in the fault free circuit. We assign a resistance value of 10 ohms to each short circuit fault.

The algorithm implemented in our own analogue circuit simulator is as follows:

Step 1. Constitute the original fault list, $\beta^0 = \{F^1, F^2, ..., F^N\}$, by inserting all possible short faults ($N=n(n\text{-}1)/2$) into the fault-free circuit to obtain N faulty versions. Here β^0 represents the original fault list and F^k is the kth fault. Let x_0^i, $i \in [0, \infty)$ be the response vector of the fault-free circuit at the ith time point and x_f^i, $i \in [0, \infty)$ be the response vector of the fth faulty circuit at the ith time point during transient concurrent fault simulation.

Step 2. Simulate all faulty circuits along with the fault-free for a user-defined number of time points. Now, x_0^i and x_f^i are available.

Step 3. Carry out a user-defined closeness measurement between each faulty circuit response, x_f^i, and the fault-free circuit response x_0^i. If a faulty circuit response is either within or far enough outside user-defined thresholds drop this fault from the fault list.

Dropped faults are not simulated further during the transient analysis. In other words it is implicitly assumed that for a given test stimulus, faults that cause sufficiently different behaviour from the fault-free are detectable and do not require further analysis, and that faults that are so far indistinguishable from the fault-free behaviour are also not worthy of further consideration. The interval over which the closeness check is performed may be chosen by the user. Clearly, it makes sense to perform the closeness check only once steady state has been reached (for a periodic stimulus). Similarly, performing a closeness check and then applying a significantly different stimulus would not be sensible as the dropped

faults would not be tested against that new stimulus. Non-convergent faults are automatically dropped.

4. EXAMPLES

We used an inverter circuit consisting of two MOS transistors and a capacitance, Fig. 1, and a 2-stage Miller opamp (from the IEEE Mixed-signal benchmark circuits suite) consisting of 9 MOS transistors, a resistance and a capacitance, Fig. 2, to validate our algorithm.

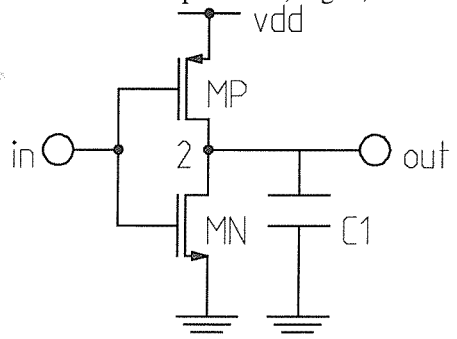


Fig.1. CMOS Inverter

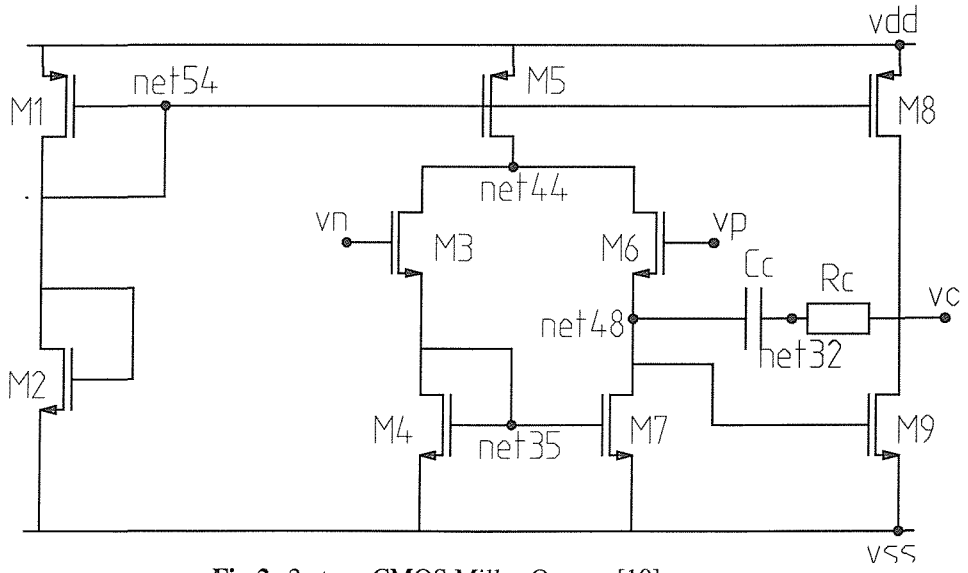


Fig.2. 2-stage CMOS Miller Opamp [10].

For the inverter circuit we used a pulse signal as the input stimulus, the single point single node Euclidean norm as the distance norm, 5% and 35% margins for closeness and "farness". The transient analysis was run over 9 periods of the input stimulus and the distance check was carried on the second period of the output signal. There are 6 possible

short faults. If we don't use fault dropping then the total number of device evaluations for during the concurrent fault simulation of these 6 faults is 23880 and the CPU time (which is obtained by doing the simulations five time and taking the average) was 4208ms. With fault dropping the total number of device evaluations was 9054, which represents 62% saving, and the CPU time was 1536ms, which is a 64% increase in the speed of the simulation. 3 faulty responses are very close to and other 3 are very far from the fault-free response so that all 6 faults are dropped from further simulation.

For the 2-stage opamp we use a sine wave as the input stimulus, and again we used the single point single node Euclidean norm where looking at the output node for the distance check. We had 56 possible short faults, but 6 of them failed DC convergence. We therefore simulated in total 50 short faults. 11 faulty responses were very close to and 38 faulty responses were very far from the fault-free response. The speed-up was 65% in terms of total number of device evaluations and 63% in terms of the CPU time.

5. Conclusions and Future Work

We have shown that using our new algorithm is effective in speeding up analogue fault simulation. We have not considered fault collapsing in this work. We will look at fault collapsing to even more increase the speed of the simulation of non-linear analogue circuits.

References

[1] Milor L.S., A Tutorial Introduction to Research on Analog and Mixed-Signal Circuit Testing, IEEE Transactions on Circuits and Systems-II, Analog and Digital Signal Processing, Vol. 45, No. 10, 1998 pp 1389-1407.

[2] Vinnakota B., Analog and Mixed-Signal Test, Prentice Hall PTR, 1998.

[3] Zwolinski M., Brown A.D., Chalk C.D., Concurrent Analogue Fault Simulation, *3rd IMSTW*, June 3-6, 1997, Seattle, Washington USA.

[4] Hou J. and Chatterjee A., CONCERT: A Concurrent Transient Fault Simulator for Nonlinear Analog Circuits, *IEEE/ACM Int. Conf on CAD*, Nov 8-12, 1998 pp 384-491.

[5] Balivada A., et al., A Unified Approach for Fault Simulation of Linear Mixed-Signal Circuits, Journal of Electronic Testing, Theory and Applications 9, 29-41 1996

[6] Abramovici, M., Breuer, M.A., Friedman, A.D., Digital Systems Testing and Testable Design, IEEE Press, 1990.

[7] Chen C.-H., Statistical Pattern Recognition, Hayden Book Company, Inc., Rochelle Park, New Jersey, 1973.

[8] Tian M.W., Shi C.-J. R., Efficient DC Fault Simulation of Nonlinear Analog Circuits, *Design, Automation and Test in Europe Conference (DATE98)*, Paris, France, Feb. 23-26, 1998.

[9] Yang Z.R. and Zwolinski M., Fast, Robust DC and Transient Fault Simulation of Nonlinear Analogue Circuits, *Design, Automation and Test in Europe Conference (DATE99)*, Munich, Germany, March 9-12, 1999.

[10] http://www.ee.washington.edu/mad/benchmarks/ benchmarks.html

8.4.5 Y. Kılıç and M. Zwolinski, "Process variation independent built-in current sensor for analogue built-in self-test", The Proceedings of 2001 IEEE International Symposium on Circuits and Systems, May 6-9, 2001, Sydney, Australia.

Process variation independent built-in current sensor for analogue built-in self-test

Yavuz Kılıç and Mark Zwolinski

Department of Electronics and Computer Science
University of Southampton
Southampton SO17 1BJ, UK
{yk, mz}@ecs.soton.ac.uk

ABSTRACT

A new design of built-in current sensor for dynamic supply current testing of analogue integrated circuits is proposed. The sensor has been designed and realized with AMS 0.8 CYE CMOS technology. The sensor occupies 0.019mm^2 silicon area, where this area is almost as big as a simple two-stage CMOS opamp. Unlike previously published sensors, this new built-in current sensor is process variation independent.

1. INTRODUCTION

Supply current monitoring has proved to be an effective method for testing digital and analogue circuits [1]-[5]. Either automatic test equipment (ATE) or Built-In Current Sensors (BICS) may be used. A BICS has advantages in terms of: equipment costs; increased testing rate; improved fault detectability; higher resolution and avoiding the influence of I/O currents [2]. Eckersall proposed using linear current mirrors to monitor analogue macros [6]. Renovell [7] added voltage monitoring to the output and integrated the two measurements over time to produce a single metric.

A number of BICS have been proposed for digital applications [2]-[4]. These require a large area for the serial active element. A suitable sensor for analogue applications was proposed in [3] and implemented in [4]. This sensor is based on a series voltage regulator, in which a series transistor is connected between the supply and the circuit under test (CUT). One drawback is the area required to realise this serial transistor, since it has to sink all the current to the CUT. A shunt voltage regulator-based BICS has been proposed in [8]. As the shunt transistor only has to sink the dynamic supply current of the CUT, the transistor area is much smaller. The main drawback with both these sensors is the requirement for a very well defined reference voltage. Moreover, both sensors are sensitive to process variations, which is a particular problem for sub-micron technologies.

2. NEW BICS SCHEME

The proposed BICS is shown in Fig.1. We are only interested in the supply current variation of the CUT rather than the absolute DC current. We monitor the dynamic current by using an active shunt element, Ms in Fig.1. The area of the shunt element depends on the current variation during normal operation. For many analogue circuits the power supply current variation is less than one tenth of the quiescent current. Therefore, the size of the shunt transistor is smaller than that of a series transistor.

Since the series element does not have to be an active device, it can be realised as a small value, small area resistance. The size of the series resistance is independent of the BICS and the CUT, unlike [8], which means it can be kept as small as possible. We could even use the parasitic resistance of the interconnect between the supply and the CUT.

The operation of the BICS is as follows: The shunt element is a PMOS transistor *Ms*, Fig.1. One input of the comparator is kept at the common mode voltage, ground for designs with dual voltage supplies. If the current drawn from the supply by the CUT changes by $\Delta Icut$ the voltage across *Rdrop* will change proportionally. This voltage is then filtered by the RC high pass network and amplified by the comparator, which regulates the voltage to the CUT. This will cause *Ms* to draw the equal but opposite current from the supply. *Ms* is operating in the linear region. Since *M1* and *M2* form a simple current mirror, *Iout* will be the same as *Is* if *M1* and *M2* are identical and have the same drain-source voltages. One way to equalize the drain-source voltages is to use the technique proposed in [3] and used in [4]. We can also use cascode current mirror in order to suppress channel length modulation effect and obtain high output impedance [9]-[10], hence increase the accuracy of the mirrored current. *Iout* can be further processed for testing purposes.

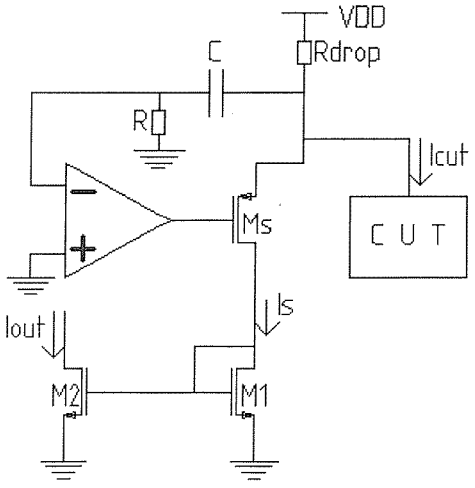


Figure 1. Process-variation independent BICS.

3. THE DESIGN OF THE PROPOSED BICS

The comparator design was based on a two stage Miller opamp. A switched capacitor offset cancellation technique,[9], was used to reduce the input offset voltage. In order not to affect normal operation, the clock frequency for the switches in the comparator is kept at least 10 times smaller than the normal CUT operation frequency. The comparator circuit schematic is given in Fig.2 and CMOS realization is given in Fig. 3.

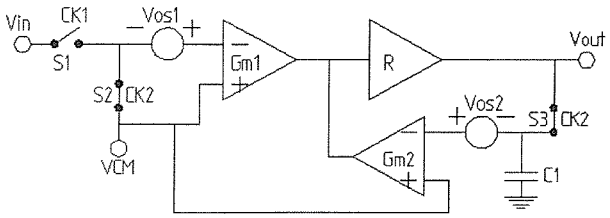


Figure 2. Switched-capacitor input offset reduced comparator.

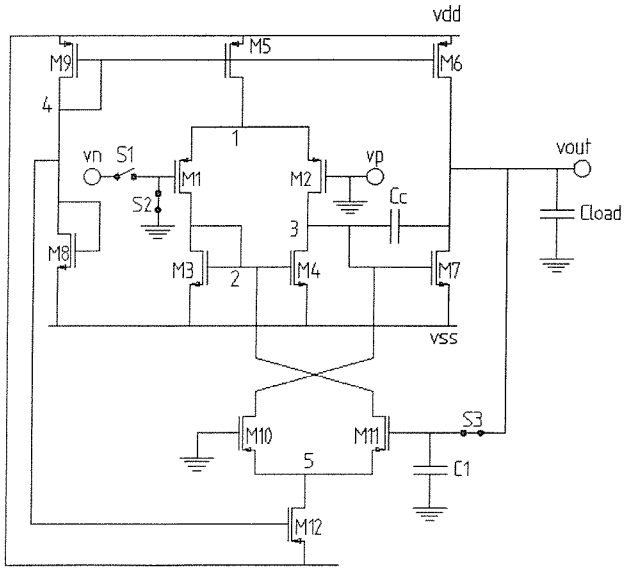


Figure 3. CMOS realization of the comparator given in Fig.2.

Switch S1 is clocked with CK1 and switches S2-S3 are clocked with CK2, where CK1 and CK2 are non-overlapping clocks in order to reduce the charge injection errors [10].

The non-overlapping clock generator proposed in [10] with a slight modification of the addition of the equal delay circuit given in [9] is used to generate equal-delay non-overlapping clocks, Fig.4. The addition of the equal-delay circuit is required since we use CMOS transmission gate based switches for S1, S2 and S3.

The comparator and the non-overlapping equal-delay clock generation circuit were designed in 0.8μm AMS CYE CMOS technology. The values of *R* and *C* depend on the normal operating frequency of the CUT. The value of *C* can be chosen to be practical for silicon integration. The resistor *R* can be implemented as a switched capacitor equivalent to give more accurate resistance and to occupy small silicon area [9].

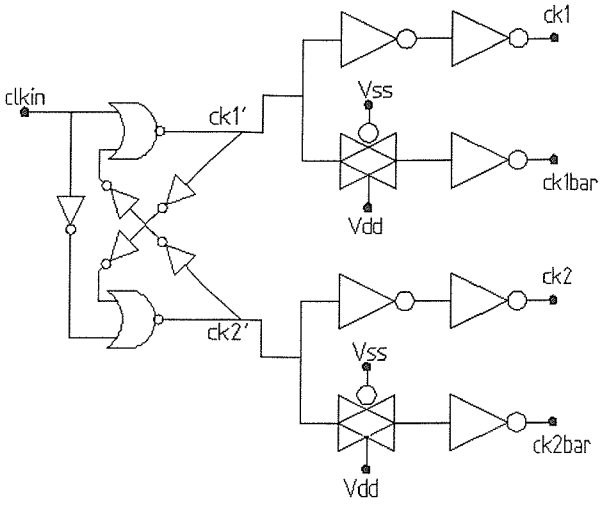


Figure 4. Equal-delay non-overlapping clock generator.

4. SIMULATION RESULTS

We used the state-variable filter, Fig. 5., from the IEEE Mixed-Signal Benchmark Suite [11] for the CUT. The filter circuit is redesigned in 0.8μm AMS CYE CMOS technology. We carried out simulations in HSPICE with BSIM 3v3 model parameters. The resistance and capacitance values are the same as the values given in [11]. Both the CUT and the BICS operate at ±2.5V.

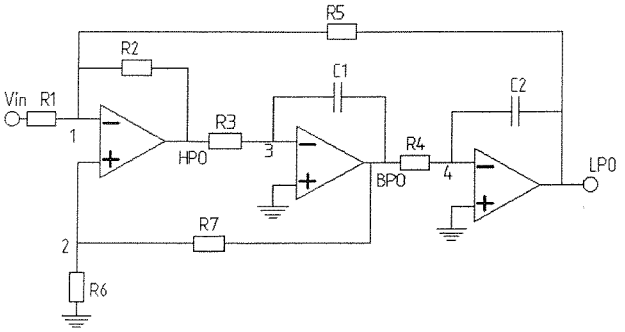


Figure 5. Continuous-time state-variable filter.

The input stimulus to the filter circuit was chosen to be a 1.3V sinusoidal wave signal at 1KHz. *R* and *C* were 100MΩ and 1pF respectively. The serial resistance (*Rdrop* in Fig.1) was 20Ω. The size of *Ms* is 2μm/1μm. To show the process independence of the BICS we carried out simulations for three different process parameter sets: worst case power (WP), typical mean (TM) and worst case speed (WS). The greatest supply voltage degradation to the CUT was 35mV for WP parameter set. This does not affect the normal operation of the CUT significantly.

The simulation results are shown in Fig.6. The dynamic current amplitude for the TM parameter set is 13.3μA, for the WP parameter set 30μA, and for the WS parameter set 3μA. As can be seen from Fig.6, the current sensed with the BICS is the same as the dynamic CUT current variation in magnitude. All the dynamic variations given above are peak-to-peak.

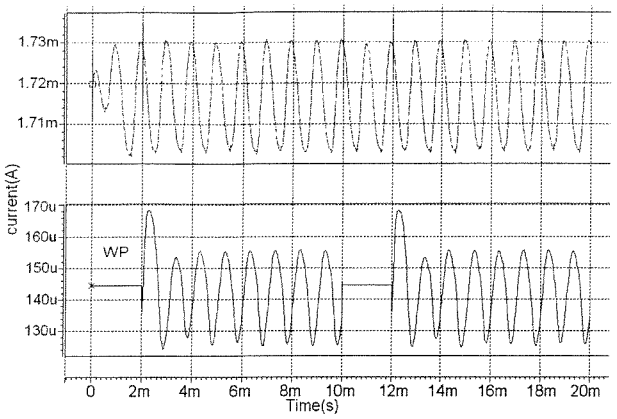


Figure 6. a) The current drawn by the CUT (top) and the monitored current with the BICS for WP parameter set.

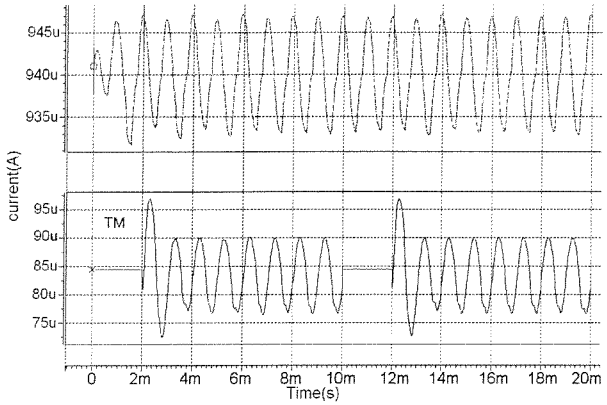


Figure 6. b) The current drawn by the CUT (top) and the monitored current with the BICS for TM parameter set.

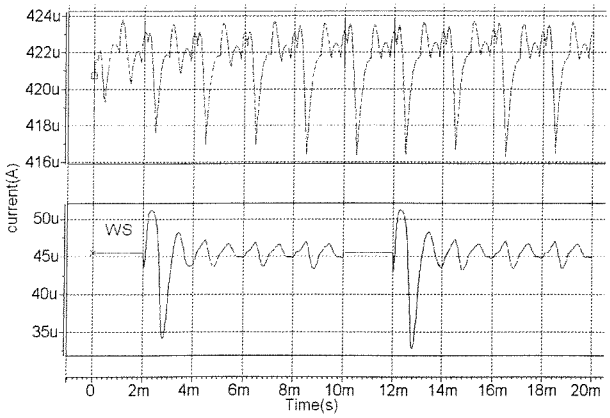


Figure 6. c) The current drawn by the CUT (top) and the monitored current with the BICS for WS parameter set.

In Fig.7 the monitored current versus comparator output voltage is given for TM parameter set. It can be seen from the figure that the output of the comparator has a $0.8V_{peak\text{-}to\text{-}peak}$ change with regard to $13.3\mu A_{peak\text{-}to\text{-}peak}$ dynamic current change due to the CUT.

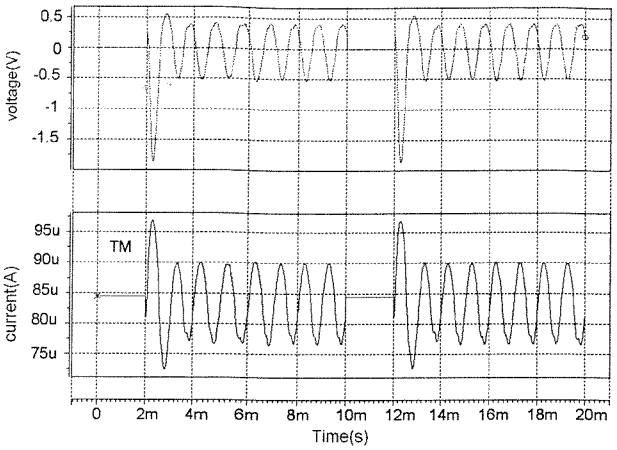


Figure 7. Monitored current versus comparator output.

We have laid out the BICS circuit and the CUT (state-variable filter) in AMS 0.8μ CYE CMOS technology. We used Cadence Virtuoso Layout editor where the layout is done in a full-custom manner. Fig.8 represents the layout of both the BICS circuit and the CUT.

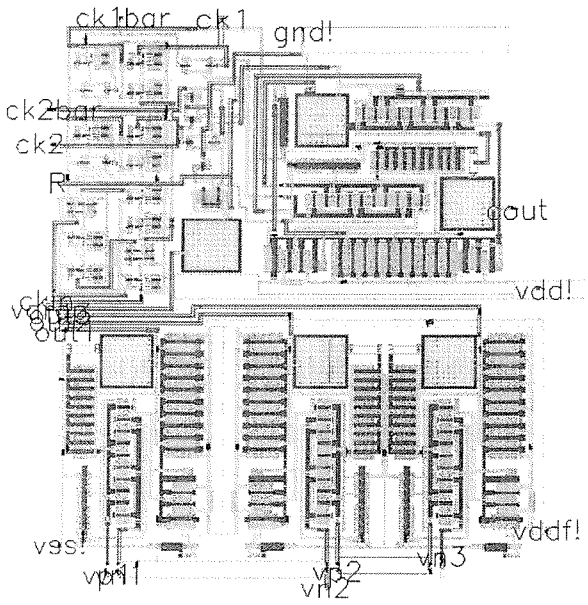


Figure 8. Layout view of the BICS circuit and the CUT.

One way to measure the correct functionality of the BICS on the fabricated IC is to measure the output voltage of the comparator instead of measuring the current on the shunt element. Therefore in Fig. 9 we give the post layout simulation results where we compare the output of the comparator with the supply voltage of the CUT.

The output of the comparator is saturating for a while because of the very small delay between the clock and its inverse, which is used to clock the CMOS switch.

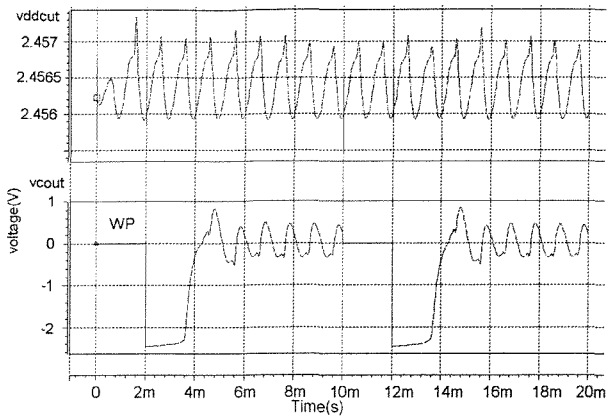


Figure 9. a) *Comparator output versus the CUT supply voltage* for WP parameter set.

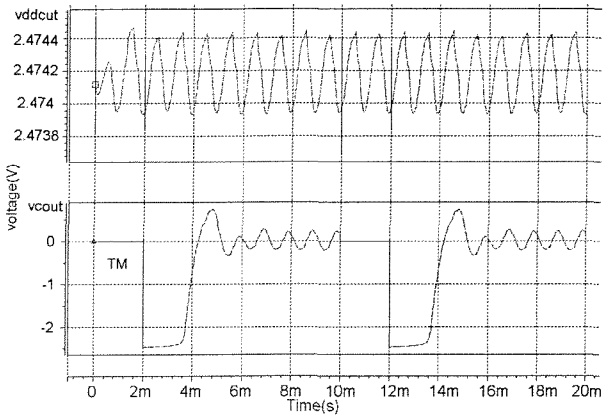


Figure 9. b) Comparator output versus the CUT supply voltage for TM parameter set.

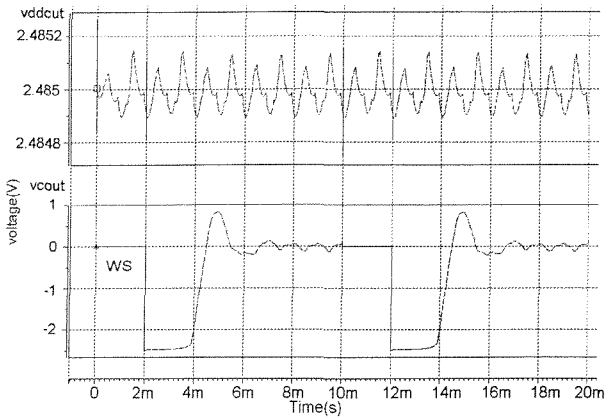


Figure 9. c) Comparator output versus the CUT supply voltage for WS parameter set.

As the comparator has a very high gain (92dB in this paper) a very small delay will be translated into a saturated voltage at the output of the comparator. As can be seen from the Fig.9, the realized BICS is functioning correctly even when the dynamic change in the supply voltage of the CUT is very small. The dynamic CUT supply change for WS parameter set (see Fig.9.c) is around $8\mu V_{peak\text{-}to\text{-}peak}$ where with this input comparator with 92dB open-loop gain causes around $0.3V_{peak\text{-}to\text{-}peak}$ change at its output. As the comparator first saturates due to the delay coming from clocks this $0.3V_{peak\text{-}to\text{-}peak}$ change seems to be masked against the saturated output value of –2.5V.

5. SUMMARY

In this paper we presented a BICS that is process-variation-independent and easily integrable for analogue built-in self-test. We have confirmed the accuracy of the BICS by HSPICE simulations. The area overhead is $0.019mm^2$ with AMS 0.8 CYE CMOS technology.

We have carried out number of measurements on the fabricated BICS IC. Measurement results confirm the correct functionality of the proposed BICS.

6. REFERENCES

[1] Y. Kılıç and M. Zwoliński, " Testing Analog Circuits by Supply Voltage Variation and Supply Current Monitoring", The 1999 IEEE Custom Integrated Circuits Conference, May 16-19, 1999, Town and Country Hotel, San Diego, California, USA.

[2] Jing-Jou Tang, Kuen-Jong Lee, and Bin-Da Liu, "A Practical Current Sensing Technique for I_{DDQ} Testing", IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Vol.3, No.2, June 1995, pp. 302-310.

[3] Seevinck M., "Analog Interface Circuits for VLSI", in Analogue IC Design: the current mode approach, 1990, Toumazou C., Lidgey F. J., and Haigh D. G. (eds.) Peter Peregrinus Ltd. IEE, London, pp. 451-489.

[4] Karim Arabi and Bozena Kaminska, "Design and Realization of an Accurate Built-In Current Sensor for On-Line Power Dissipation Measurement and I_{DDQ} Testing", International Test Conference, 1997, pp.578-586.

[5] Zwolinski M, Chalk C., Wilkins B.R. and Suparjo B.S., "Analogue Circuit Test using RMS Supply Current Monitoring", 2nd IEEE International Mixed Signal Testing Workshop, June 1996.

[6] Eckersall K.R., Wrighton P.L., Bell I.M., Bannister B.R., and Taylor G.E., "Testing Mixed Signal ASICs through the use of Supply Current Testing", Proc. 3rd European Test Conf., Rotterdam, The Netherlands, April 1993, pp.385-391.

[7] Renovell M., Azais F., and Bertrand Y., "On-chip signature analyser for analogue circuit testing", Electronics Letters, 1996, Vol.32, No.24, pp. 2185-2186.

[8] Y. Kılıç, C.D. Chalk and M. Zwolinski, "Design and Realisation of a new Built-In Current Sensor for Mixed-Signal IDDD Test", 5th IEEE International Mixed Signal Testing Workshop, Vancouver (Whistler), Canada, June 15-18, 1999.

[9] "Design of Analog CMOS Integrated Circuits", Behzad Razavi, McGraw-Hill, 2000.

[10] "Analog Integrated Circuit Design", David A. Johns, Ken Martin, John Wiley & Sons, Inc., 1997.

[11] http://www.ee.washington.edu/mad/benchmarks/benchmarks.html

8.4.6 Yavuz Kılıç and Mark Zwolinski, "Speed-up Techniques for Fault-based Analogue Fault Simulation", The IEEE European Test Workshop, May 29 - June 01, 2001, Stockholm, Sweden.

Speed-up Techniques for Fault-based Analogue Fault Simulation

Yavuz Kılıç[1,2] and Mark Zwoliński[1]
{yk, mz}@ecs.soton.ac.uk[1], Yavuz.Kilic@philips.com[2]

Abstract

In this paper, new approaches to speed up dc and transient analogue fault simulation at transistor level are discussed. The effectiveness of the techniques is evaluated through the use of IEEE Mixed-Signal Benchmark Circuits. Up to 19 times reduction in CPU time and 100% fault coverage is possible for the benchmark opamp circuit.

1 . Introduction

Analogue fault simulation is the first step to fault coverage analysis, fault grading, fault collapsing, and BIST [1]. Fast fault simulation, in general, is crucially important in terms of speeding up the entire testing process.

There are still not many analogue fault simulation algorithms, in contrast to digital fault simulation. This is partly because there is not yet a standard fault model for analogue circuits, such as the stuck-at fault model in digital circuits. Without a fault model, concepts such as fault cover and fault detectability are difficult to quantify. Therefore the design of fault simulation algorithms that exploit such information is difficult.

Nevertheless, it is generally acknowledged that analogue fault simulation is slow. To make analogue fault simulation usable, new techniques need to be developed to cut the CPU time. There has been some research on new algorithms for speeding up analogue fault simulation [2], [3], [4].

In [3], a concurrent fault simulator (CONCERT) for nonlinear analogue circuits is presented. In order to speed up the simulation process, the authors use fault ordering, state prediction and reduced-order fault matrix computation.

In [4], the authors consider only linear mixed-mode circuits. They attempted to represent the analogue blocks in the sampled Z-domain using discrete-time state equations in order to provide a common framework for the simulation of both analogue and digital blocks.

The objective of the work described in this paper is to develop techniques to speed up analogue fault simulation. The rest of the paper is organised as follows. In Sec. 2 a new closeness measurement technique is developed for fault dropping in DC and transient analyses. Based on the closeness measurement developed in Sec. 2, a new analogue concurrent fault simulation algorithm is discussed in Sec. 3. Sec. 4 presents the speed-up results obtained in analogue fault simulation applying the developed techniques to a set of benchmark circuits. Conclusions and possible future research is discussed in Sec. 5.

2 . Closeness Measurement

There are number of closeness measurements available in the literature but they are not easily applicable to the problem of analogue fault simulation [6]. In the analogue domain, the most widely used closeness measurement has been the Euclidean distance [7], [8].

In [7], the normalised absolute distance is used. The single point closeness measurement is adopted in order to detect the similarity between faulty responses and the fault-free response in DC fault simulation.

In [8], it is suggested that taking a single point closeness measurement into account might not accurately detect the distance between the fault-free circuit response and faulty circuit responses, as the distance might not always be the same during the simulation. Therefore a multi point closeness measurement for the DC fault simulation of analogue circuits is used.

In the multi-point closeness measurement, let x_f be the vector containing the node voltages and/or branch currents of the faulty circuit and x_g be the vector of the same elements for the fault-free circuit. The distance between the vectors x_f and x_g is given as follows [8]:

$$d_{fg} = \frac{1}{M}\sum_{m=1}^{M}\left\| x_g^m - x_f^m \right\| \qquad (2\text{-}1)$$

where d_{fg} is the distance between the vectors x_f and x_g. x_g^m and x_f^m are the responses for the good circuit and the f^{th} faulty circuit at the m^{th} N-R iteration. M is the number of consecutive N-R iterations for the closeness measurement.

This closeness measurement is used to decide whether the faulty circuit response is close enough to the fault-free circuit response within a specified threshold to drop this fault from the fault list at the succeeding iteration step. If a faulty response is far enough from the fault-free response within a specified threshold we then can drop this fault also from the fault list, as this fault is deemed detectable. We use the same closeness measure with a different threshold to determine this distance between faulty and fault-free responses.

In this work we consider both DC and transient fault dropping. For the DC fault simulation we propose to do the closeness check at the end of the DC operating point calculation of all faulty circuits along with the fault-free one. This is because the CPU time spent on DC analysis is very small compared with the CPU time spent on transient analysis. For transient fault simulation, we propose to do the closeness check using simulated values obtained after a number of time points.

Assuming that x_f is the vector containing the node voltages and/or branch currents of the faulty circuit and x_g is the vector of the same elements for the fault-free circuit, the distance between the vectors x_f and x_g is now given as:

$$d_{fg} = \frac{1}{n}\sum_{i=1}^{n}\left\| x_g^i - x_f^i \right\| \qquad (2\text{-}2)$$

[1] Dept. of Electronics and Comp. Science, University of Southampton
Southampton SO17 1BJ, UK

[2] Philips Semiconductors, Southampton
SO15 0DJ, UK

where d_{fg} is the distance between the vectors x_f and x_g. x_g^i and x_f^i are the responses for the good circuit and the k^{th} faulty circuit at the i^{th} time point and n is the number of consecutive time points for the closeness measurement.

In this paper, both normalised absolute and relative Euclidean norms for doing the closeness check are discussed. We also allow a choice between those two distance measures with different options: single point single node, single point multi-node, multi-point single node, multi-point multi-node. These options can be directly entered with the use of ".options" command from the SPICE netlist. Note that for the DC analysis only single-point multi-node distance checks are meaningful as we do these checks at the end of the DC analysis.

2.1 Threshold Calculation

Let a_f be the node voltage and/or branch current of a faulty circuit and a_g be the same value for the fault-free circuit in the case of the single node closeness check. If we assume that the closeness measure is the relative Euclidean norm and $a_f = ka_g$, then the threshold is given as:

$$th = \|a_g - a_f\| = \|a_g - ka_g\| = |1-k|a_g \qquad (2\text{-}3)$$

where th is the threshold between the faulty value and the fault-free value.

If the closeness measure is the normalised absolute Euclidean norm then the threshold is:

$$th = \left|\frac{a_g - a_f}{a_g}\right| = \left|\frac{a_g - ka_g}{a_g}\right| = |1-k| \qquad (2\text{-}4)$$

In the case of the multi-node closeness check, each element of the faulty vector might not be perturbed by the same amount (k) as a result of the fault being injected in the fault-free circuit. There will be a vector k consisting of elements (k_1, k_2, ... , k_l) where l is number of nodes in the circuit being simulated. In order to simplify the calculations, however, we assume that $k_1=k_2=...=k_l=k$.

Now we can say that we want to drop the fault when the faulty response is within the 5% of the fault-free response or 35% far from the fault-free response. In this paper we assume that if a faulty response is within 5% of the fault-free response or 35% far from the fault-free response then the fault can be dropped from further analysis.

3 . Concurrent Fault Simulation

Concurrent fault simulation is well understood for digital circuits [5]. Recently, the principle of concurrent fault simulation has been applied to analogue circuits [2], [3]. In concurrent transient analogue fault simulation, all the faulty circuits along with the fault-free circuit are simulated simultaneously at a time point before simulation proceeds to the next time step. In this paper we use different time steps to simulate each faulty version of the circuit along with the fault-free one, in which case the simulation will speed up since some faulty versions might take fewer time steps to simulate. This is distinct from the work done in [3] and [8]. Further, if the terminal value of the faulty device is close enough to the terminal value of the fault-free device within a specified threshold then the fault-free device values can be reused for the faulty one, thus reducing the CPU time.

3.1 Concurrent Fault Simulation Algorithm

Here, we have used structural short circuit faults, which are constructed from the fault-free circuit netlist. If the fault free circuit has n nodes then we have $n(n\text{-}1)/2$ faulty versions of the circuit to analyse as we assume that there is a short circuit fault between each pair of nodes in the fault free circuit. We assign a resistance value of 10 ohms to each short circuit fault. Similarly the work can easily be extended to cover open circuit faults as well.

The steps for the algorithm that has been implemented in our own analogue circuit simulator in C language are as follows:

Step 1. Constitute the original fault list, $\beta^0 = \{F^1, F^2, ..., F^N\}$, by inserting all possible short faults ($N=n(n\text{-}1)/2$) into the fault-free circuit to obtain N faulty versions. Here β^0 represents the original fault list and F^k is the kth fault. Let x_0^{DC} be the response vector of the fault-free circuit and x_f^{DC} be the response vector of the fth faulty circuit at the at the end of the concurrent DC simulation, x_0^i, $i \in [0, \infty)$ be the response vector of the fault-free circuit at the ith time point and x_f^i, $i \in [0, \infty)$ be the response vector of the fth faulty circuit at the ith time point during concurrent transient fault simulation.

Step 2. DC simulate all faulty circuits along with the fault-free one. Now x_0^{DC} and x_f^{DC} are available.

Step 3. Carry out a closeness measurement between each faulty circuit response, x_f^{DC}, and the fault-free circuit response x_0^{DC}. If a faulty response is either within or far enough outside user-defined thresholds then drop this fault from the original fault list. Now we are left with the faults to be taken into account for the concurrent transient simulation.

Step 4. Simulate all remaining faulty circuits along with the fault-free one for a number of time points. Now, x_0^i and x_f^i are available.

Step 5. Carry out a closeness measurement between each faulty circuit response, x_f^i, and the fault-free circuit response x_0^i. If a faulty circuit response is either within or far enough outside user-defined thresholds then drop this fault from the fault list.

Dropped faults are not simulated further during the transient analysis.

There are some faults that cause the simulator not to converge at the DC initial solution. These faults therefore cannot be dealt with and hence are dropped from the fault list.

4 . Examples

We used a simple CMOS inverter circuit, and the 2-stage Miller opamp, the state-variable active filter, and the leapfrog filter, from the IEEE Mixed-Signal Benchmark Circuits suite [9], in order to validate our techniques.

For the CMOS inverting amplifier we used a sine wave as the input stimulus. We used all possible norms, where looking at the output node for the distance check. A distance check was carried out on the second period of the output signal where we did simulations for 10 periods of the input stimulus. We had 56 possible short faults, but 8 of them failed DC convergence. We therefore simulated in total 48 short faults. 5% and 35% margins for closeness and "farness" are used in order to drop a fault. The speed-up (in terms of the CPU time and the number of device evaluations) and fault coverage for different norms over simulated faults are presented in Table I.

For the above-mentioned benchmark circuits we used single-point multi-node Euclidean norm as the distance norm, 5% and 35% margins for closeness and "farness". The transient analysis was run over 10 periods of the input stimulus and the distance check was carried on the second period of the output signal, where we assume that after the first period of the input stimulus the steady state is reached. Note that the choice of which norm to use and how to evaluate the results so that to drop a fault from further consideration is totally user-defined.

Two implications of the results we obtain are: fault coverage and speed-up in the CPU time. In Table II, we present these results for different benchmark circuits.

5 . Conclusions and Future Work

We have shown that using our new techniques are effective in speeding up analogue fault simulation. By using the techniques proposed in this paper, one can achieve over 19 times speed-up in CPU time while 100% fault coverage is also possible for the benchmark opamp circuit. We have not considered fault collapsing in this work. We will look at fault collapsing to increase even more the speed of simulation of non-linear analogue circuits.

6 . Acknowledgements

This work has been supported by EPSRC grant GR/L35829/01. The authors would also like to thank Philips Semiconductors, Southampton for its financial support for this paper.

7 . References

[1] B.Vinnakota, "Analog and Mixed-Signal Test", Prentice Hall PTR, 1998.

[2] M. Zwolinski, A. D. Brown, C. D. Chalk, "Concurrent Analogue Fault Simulation", 3rd IMSTW, June 3-6, 1997, Seattle, Washington USA.

[3] J. Hou and A. Chatterjee, "CONCERT: A Concurrent Transient Fault Simulator for Nonlinear Analog Circuits", IEEE/ACM Int. Conf on CAD, Digest of Technical Papers, p 384-491, Nov 8-12, 1998.

[4] A. Balivada, et. al., "A Unified Approach for Fault Simulation of Linear Mixed-Signal Circuits", Journal of Electronic Testing, Theory and Applications 9, 29-41 (1996), 1996 Kluwer Academic Publishers, manufactured in The Netherlands.

[5] Abramovici, M., Breuer, M.A., Friedman, A.D, "Digital Systems Testing and Testable Design", IEEE Press, 1990.

[6] Chi-hau Chen, *Statistical Pattern Recognition*, Hayden Book Company, Inc., Rochelle Park, New Jersey, 1973.

[7] Michael W. Tian, C.-J. Richard Shi, "Efficient DC Fault Simulation of Nonlinear Analog Circuits", Design, Automation and Test in Europe Conference (DATE98), Paris, France, Feb. 23-26, 1998.

[8] Z. R. Yang and M. Zwolinski, "Fast, Robust DC and Transient Fault Simulation of Nonlinear Analogue Circuits", Design, Automation and Test in Europe Conference (DATE99), Munich, Germany, March 9-12, 1999.

[9] http://www.ee.washington.edu/mad/benchmarks/benchmarks.html

Table I. Fault simulation results for inverting opamp.

opamp	Simulation Time		Speed-up		Fault Coverage					
FD method	CPU (s)	Tot # of Dev. Ev.	CPU	Dev. Ev.	Total # of sim. faults	# of close faults (DC)	# of far faults (DC)	# of close faults (TR)	# of far faults (TR)	Fault Coverage (%) (DC+TR)
seucl snode	8.19	13644	16.8	19.3	48	14	28	0	6	100
seucl mnode	22.1	40086	6.2	6.6	48	14	28	0	0	87.5
meucl snode	8.08	13644	16.9	19.3	48	14	28	0	6	100
meucl mnode	22.22	40086	6.2	6.6	48	14	28	0	0	87.5
snabsl snode	26.95	48897	4.6	5.4	48	19	18	1	5	89.6
snabsl mnode	40.76	74358	3.4	3.5	48	19	18	0	0	77.1
mnabsl snode	21.84	40707	6.3	6.5	48	19	18	0	8	93.8
mnabsl mnode	18.87	37822	7.3	7	48	19	18	0	9	95.8
NO FD	137.23	263151								

Table II. Speed-up and Fault coverage for benchmark circuits.

	Speed-up		Fault Coverage					
Benchmark	CPU	Dev. Ev.	Total # of simulated faults	# of close faults (DC)	# of far faults (DC)	# of close faults (TR)	# of far faults (TR)	Fault Coverage (%) (DC+TR)
INV	5.4	5.2	6	4	2	0	0	100
OPAMP	6.2	6.6	48	14	28	0	0	87.5
FILTER	6	5.7	95	35	34	1	6	80
LEAPFROG	5.4	5	378	176	97	0	5	73.5

8.4.7 Yavuz Kılıç and Mark Zwolinski, **"BEHAVIOURAL/MACRO MODELLING TO SPEED-UP ANALOGUE FAULT SIMULATION", International Conference on Electrical and Electronics Engineering (ELECO 2001), Bursa, Turkey, 7-11 November 2001.**

BEHAVIOURAL/MACRO MODELLING TO SPEED-UP ANALOGUE FAULT SIMULATION

Yavuz Kılıç (Member IEEE) and Mark Zwoliński[1] (Senior Member IEEE)
e-mail: Yavuz.Kilic@philips.com
Philips Semiconductors, Southampton, UK

Keywords: macromodelling, behavioural modelling, VHDL-AMS, fault simulation

Abstract

The main difficulty in generating test patterns for analogue and mixed-signal circuits is fault simulation. Analogue fault simulation is much slower than the digital equivalent. Two of the techniques to speed up the analogue fault simulation process are: fault dropping/collapsing, in which faults that have similar circuit responses compared with the fault-free circuit response and/or with another faulty circuit response are considered equivalent; and behavioural/macro modelling, whereby parts of the circuit are modelled at a more abstract level, therefore reducing the complexity and the simulation time. This paper discusses behavioural/macro modelling in order to speed-up fault simulation for analogue circuits.

I. Introduction

As transistor sizes keep shrinking, integrated circuits (ICs) have been growing in size and complexity. This growth in ICs causes testing to be much more difficult. For digital circuits the problem of testing can be simplified by using standard fault models and fast fault simulation. Faults in digital circuits can be modelled as stuck-at, bridging and open faults. These structural faults can then be used to generate functional test vectors. The objective in developing a test program for a digital circuit is to determine whether or not a fault exists using the smallest possible number of test vectors [2]. Therefore, test pattern generation is the process of selecting an optimal set of tests from all possible input patterns. This optimal test pattern selection can be done in an ad-hoc manner for small and simple circuits. For larger circuits the optimal set of tests can be chosen using algorithms such as the D-algorithm or PODEM [2].

A test pattern is evaluated by looking at its fault coverage. All faults detected with this pattern can be dropped from further consideration. Fault simulation is done for the assessment of the fault coverage. There are number of fault simulation techniques for digital circuits. Serial fault simulation is perhaps the simplest method. For each fault, a copy of the circuit with the fault inserted into it is created. Then, all the faulty copies of the circuits along with the fault-free original are simulated with the given test patterns. If the output of a faulty circuit differs from the fault-free output, that fault is considered to be detectable.

Another fault simulation technique for digital circuits is concurrent fault simulation [2]. The differences between the faulty and the fault-free circuit behaviours might be relatively small. Therefore, in concurrent fault simulation the aim is to avoid redundant element evaluation when the fault-free and faulty behaviours are the same, hence reducing the computational effort.

Analogue and mixed-signal fault simulation has been limited to the serial technique. Faster methods are not easily applied to analogue and/or mixed-signal circuits, because faults do not affect the circuit behaviour in a binary manner.

One way to speed-up fault simulation for analogue and mixed-signal circuits is to use behavioural or macro models, where parts of the circuit are modelled at a more abstract level, reducing the complexity and hence the simulation time. In this paper we summarise research in behavioural/macro modelling for speeding up analogue fault simulation. The structure of the paper is as follows. First, macromodelling for analogue circuits is presented. Then behavioural modelling is discussed with a case study. In section IV, behavioural modelling using Hardware Description Languages (HDLs) is summarised. In secion V, a behavioural fault model is developed in VHDL-AMS for an opamp circuit operating in inverting amplifier configuration. Finally, in section VI some conclusions are drawn.

[1] University of Southampton, Department of Electronics and Computer Science, Southampton, UK, SO17 1BJ

II. Macromodels for Analogue Circuits

Simulation at the transistor level for analogue circuits is computationally very expensive. Therefore, one way to reduce this high simulation cost is to partition a large analogue circuit into smaller functional blocks such as opamps and replace each functional block with its *macromodel* or to describe each block using mathematical equations (a *behavioural model*). This technique is sometimes called *hierarchical simulation* [3].

The word *macromodel* usually refers to a compact representation of a circuit that captures those features that are useful for a particular purpose while discarding redundant information [4]. Macromodels developed for SPICE-like simulators are basically electrical networks containing devices such as voltage-controlled voltage sources instead of the full transistor network and with fewer nodes than the original circuit.

Many circuits are designed in a modular style, in which functional units are connected to achieve the design specifications. The behaviour of the whole circuit is determined by how the individual units interact with each other, while what happens inside each is unimportant in terms of the behaviour of the entire circuit. The accuracy of a macromodel must, therefore, be defined in terms of how closely its input-output behaviour matches that of the original unit [4].

Since the early 1970s, a number of macromodels have been developed, mainly for integrated operational amplifier circuits (opamps) [3]-[14]. Boyle et al developed a macromodel for integrated bipolar opamp circuits [5]. This macromodel was six times less complex (in terms of the node count) than the original opamp circuit, and the simulation time was an order of magnitude faster than the device-level model.

The derivation of component values for the Boyle macromodel is not, however, straightforward. Some parameters are modelled using unbalanced input devices and other parameters interact. Therefore, a modular approach was suggested [8], in which a macromodel was derived simply from the published data sheets. Individual parameters were modelled separately and the results combined to provide the output response. Since the parameters were separated they did not interact and only those required were included.

Recent research has focused on how to capture the effect of a fault in an analogue circuit within its macromodel [1], [3], [15]. In, [3] the fault macromodelling problem was formulated in terms of deriving the macro parameter set, B, based on the performance parameter set, P, (gain, bandwidth, samples on the frequency or time response curves, etc.) of the transistor-level faulty circuit. The accuracy of the macromodel was evaluated by checking the consistency of the performance parameter set, P, between the transistor-level circuit and the macromodel.

Two steps are needed to obtain the macromodel for a functional block within an analogue circuit [3]:

1. Perform transistor level fault simulation for each faulty circuit to obtain the value of the performance parameter set P
2. Map each performance parameter set P to the corresponding macro parameter set, B. This is referred to as *parameter mapping*.

It was assumed that the transistor-level fault list is given and the macromodel structure and the performance parameter set, P, to be matched are predetermined by the circuit designer.

There are several ways to do parameter mapping. One simple approach is based on analytical design equations that express the macro parameter set, B, as analytical functions of the performance parameter set, P, and the value of B is derived by function evaluation. As analogue ICs get more complex, this approach is becoming more difficult. Another simple approach is to build an empirical mapping function, $B=F(P)$, based on a large number of data pairs (P, B), referred to as the *training set* [3]. Usually the training set is generated by randomly selecting M out of the N performance parameter sets for the faulty circuits obtained by transistor-level simulation and then the value of the macro parameter set B for each selected P is derived. The derivation of each data pair usually requires multiple runs of macromodel-level simulation [3].

Macromodelling in general and fault macromodelling in particular, using SPICE-like languages, nevertheless, have been shown to be very difficult [1], [3]-[18]. Therefore, another easier and perhaps more efficient way of modelling analogue circuits at a higher level is necessary.

III. Behavioural Modelling

A behavioural model describes a circuit block in terms of mathematical equations modelling the functionality of the block, for example, in terms of the input-output relationship. Behavioural modelling has been used for speeding up analogue simulation in general [19] and analogue fault simulation in particular [1], [15], [18], [20]. In [19], analogue circuits were modelled behaviourally in the C programming language. Broyden's method was used to formulate and solve the model equations in a custom simulator. Broyden's method was originally proposed in [21] as an algorithm for the solution of systems of nonlinear equations, i.e. of the derivatives of a set of functions. The main drawback with the work described in [19] is that since the technique does not require derivatives it cannot be used for small-signal analysis.

In [15], Chang et al presented a behavioural fault model derived from a macromodel of a CMOS operational

amplifier from the IEEE Mixed-Signal Benchmark Suite [22] (Figure 1).

The faulty macromodel was developed using DC-sweep analysis. The DC behaviour of the benchmark opamp operating in inverting, non-inverting and unity gain amplifier configurations was first investigated under different faulty conditions, as shown in Figure 2. Single transistor catastrophic faults, bridging/short and nearly open faults, and parametric faults with W (channel width), L (channel length) and V_t (threshold voltage) varied by ±10% were used for each transistor. Then an attempt was made to group the different faulty behaviours. By comparing the fault-free offset voltage measured at the inputs of the opamp operating in one of the three configurations with the equivalent faulty circuits, four different equivalent fault types were derived [15]: M4 drain-to-gate short (Type I), M5 drain-to-source short (Type II), M7 drain open (Type III), and M5 drain-to-source short (Type IV). The first three fault types existed for the opamp operating in the inverting configuration; the Type IV fault group was found for the non-inverting configuration.

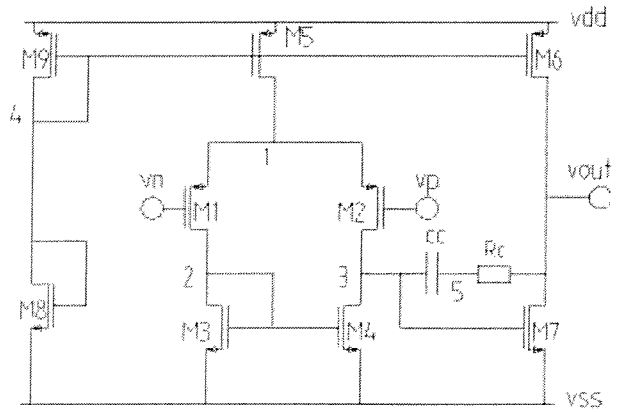


Figure 1. The 2-stage CMOS Miller opamp used in [15] for behavioural fault modelling.

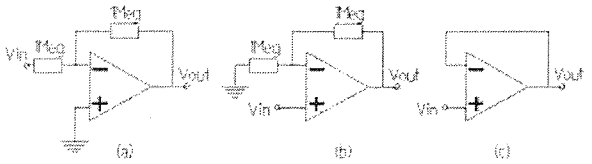


Figure 2. Three different configurations used in [15] for the benchmark circuit given in [22]: (a) Inverting amplifier, (b) non-inverting amplifier, and (c) unity gain buffer.

The input offset voltage (measured between the non-inverting and inverting inputs of the opamp in the closed-loop configurations) and the output voltage versus the input voltage for the fault-free opamp operating in the three configurations were determined by HSPICE simulations and are shown in Figure 3, Figure 4, and Figure 5, respectively.

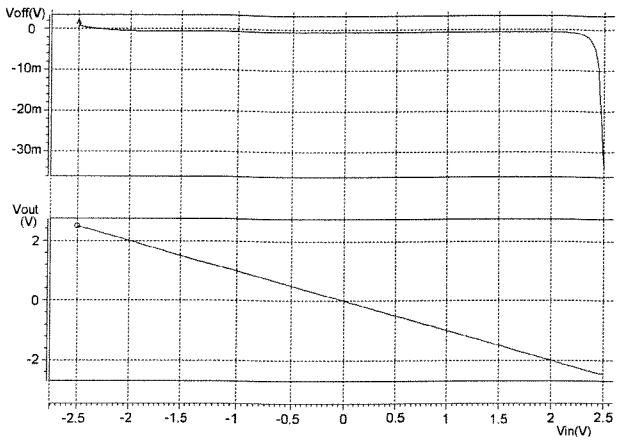


Figure 3. Input offset voltage and output voltage versus input voltage for the fault-free inverting amplifier.

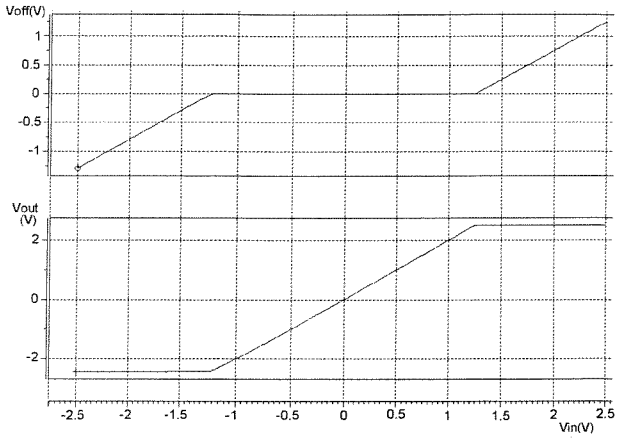


Figure 4. Input offset voltage and output voltage versus input voltage for the fault-free non-inverting amplifier.

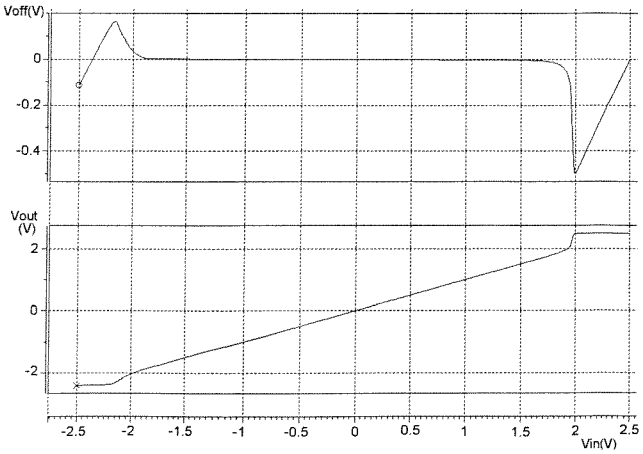


Figure 5. Input offset voltage and output voltage versus input voltage for the fault-free unity gain buffer.

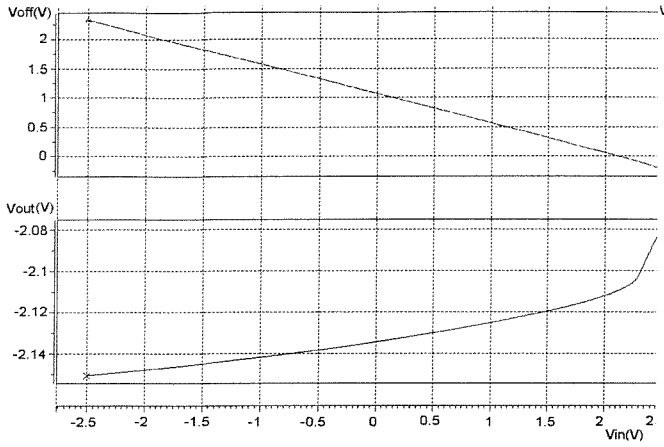


Figure 6. Input offset voltage and the output voltage for the Type I fault.

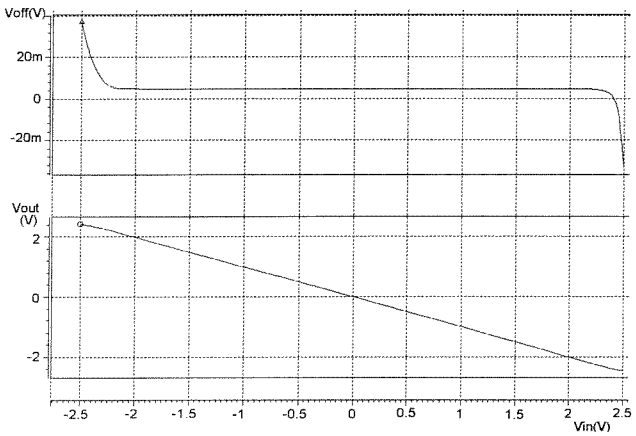


Figure 7. Input offset voltage and the output voltage for the Type II fault.

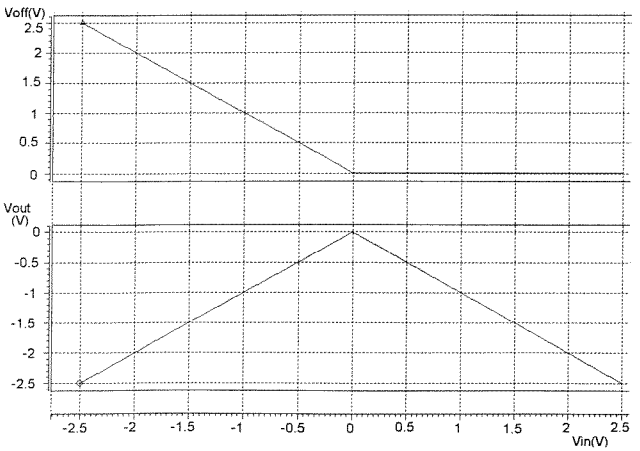


Figure 8. Input offset voltage and the output voltage for the Type III fault.

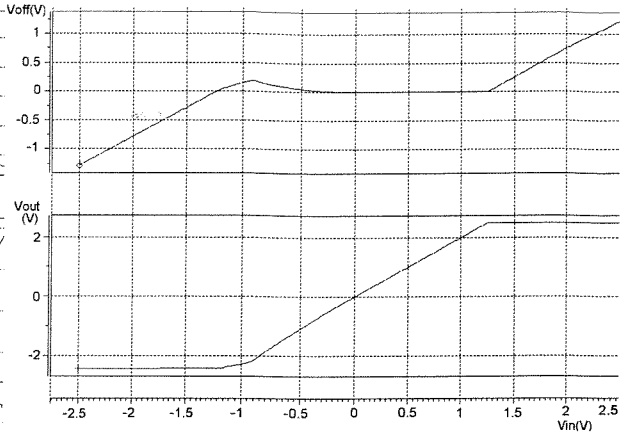


Figure 9. Input offset voltage and the output voltage for the Type IV fault.

The input offset voltage and the output voltage for each fault group with respect to the input voltage were also found by HSPICE simulations and are shown in Figure 6, Figure 7, Figure 8, and Figure 9, respectively.

As can be seen from Figure 6 and Figure 9 responses obtained for Type II and Type IV faults are quite similar to the fault-free responses in Figure 3 and Figure 4. Type II and Type IV input offset voltages are somewhat different from the fault-free responses. The input offset voltage has a small DC level for Type II faults, but has a non-linear characteristic for Type IV faults.

The remaining two faults have very different characteristics to the fault-free equivalents for both input offset voltages and output voltages. It can be concluded from the figures that a Type I fault causes the inverting amplifier output to be nearly stuck at a negative voltage near to the negative supply voltage level. A Type III fault causes the inverting amplifier output to have a non-inverting characteristic for the negative values of the DC input signal, and an inverting characteristic for the positive values of the DC input signal. As can be seen from the figures above, the input offset voltage at the inputs of the opamp has a linear characteristic for Type I faults, and a piecewise linear characteristic for Type III faults.

The macromodel given in Figure 10 for the inverting opamp was used to derive the input output relationship under fault conditions [15]:

$$V_{out} = A_{CL}\left[(1+m)V_{in} + k\right] \tag{1}$$

where A_{CL} is the closed-loop gain for the opamp. The parameters m and k are given in [15] as:

$$m = \frac{-R2}{D+R2} \tag{2}$$

and

$$k = aVos + bVdd + cVss \tag{3}$$

where

$$D = B(R2 \,//\, Ro \,//\, Rdd \,//\, Rss),$$

$$B = \left(\frac{A}{Ro} - \frac{1}{R2}\right)(Rid \,//\, R1 \,//\, R2 \,//\, 2Ricm),$$

$$a = \frac{R2 \,//\, D}{A_{CL}(R1 \,//\, R2 \,//\, 2Ricm \,//\, BR2)},$$

$$b = \frac{SF}{A_{CL} Rdd},$$

$$c = -\frac{SF}{A_{CL} Rss},$$

$$A_{CL} = -\frac{R2}{R1},$$

$$SF = Rdd \,//\, Rss \,//\, Ro \,//\, (R2 \,//\, R11) \,//\, \frac{Ro(R11 \,//\, R2)}{AR11}$$

$$R11 = R1 \,//\, 2Ricm \,//\, Rid,$$

and A represents the open-loop gain.

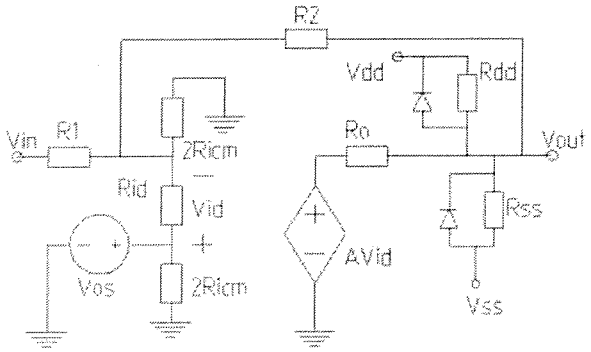


Figure 10. Macromodel used in [15] to derive the input-output relationship for the closed loop inverting opamp.

The non-ideal effects such as the input offset voltage, *Vos*, the finite open-loop gain, *A*, and the finite input and output resistances, *Rid* (differential mode input resistance), *Ricm* (common mode input resistance), *Ro* (output resistance), and the resistances from output node to the supply rails (*Rdd* and *Rss*) to model output stuck-at faults were taken into account in deriving equation (1). Note that for the fault-free case, *Rid*, *Ricm*, *Rdd*, *Rss*, and *A* would be infinite, *Vos*, and *Ro* would be zero, hence $m \to 0$, and $k \to 0$. When a fault causes the output to be stuck at some voltage level, $D \to 0$, therefore $m \to -1$, and k is the value of the stuck output voltage; the closed-loop gain, A_{CL}, is assumed to be unity. As they are dealt with elsewhere [15], the derivation of the above equations will not be given here.

In [15], the current limiting effect was also modelled. This is due to the finite supply voltage at the output of the opamp. It is claimed that the model covers all the parametric faults and 92.5% of the catastrophic faults that were considered. The model could not model M4 drain-to-gate short, M5 drain-to-source short, M1 open-gate faults for the non-inverting amplifier and M2 drain-to-gate short, M4 drain-to-gate short, M5 drain-to-source short, M1 open gate, M3 open source and M5 open gate faults for the unity gain buffer.

IV. Behavioural modelling using HDLs

HDLs have been in use for behavioural modelling and simulation of digital circuits as well as analogue electronic systems, fluid concentrations in chemical processes, and parachute jumps since 1960 [26]. Currently two of the most widely used standards for modelling digital designs are VHDL [23], and Verilog [24]. For analogue circuits, the choice has been between SPICE and proprietary analogue HDLs.

Analogue HDLs support the description of systems of differential and algebraic equations (DAEs). The solution of these systems varies continuously with time. Most analogue HDLs support both structural composition and conservation semantics, in addition to behavioural descriptions. Examples of such languages are FAS [27] SpectreHDL [28] and Verilog-A [29].

Mixed-signal design has depended on the use of separate HDLs for the analogue and digital parts or, again, on proprietary languages. Mixed-signal languages support both event-driven techniques and differential and algebraic equations in one simulator. Simulators in this category are MAST/Saber [30], VeriasHDL [30], AdvanceMS [27], Hamster [31].

Both VHDL and Verilog have been extended to analogue and mixed-signal design: VHDL-AMS [32], and Verilog-AMS [29]. The analogue extensions to VHDL and Verilog should alleviate the multiple-language problem [25].

Since VHDL-AMS was standardised in 1999 there has been some work done on fault modelling using VHDL-AMS. One reason for the limited progress is perhaps that there is not yet a robust VHDL-AMS simulator available that has all the VHDL-AMS constructs, such as procedural statements, implemented. Perkins et al attempted to use analogue VHDL for fault modelling and simulation with very limited success [1]. The authors used the HDL-A modelling language with the ELDO simulator from Anacad. Behavioural model simulation using HDL-A and ELDO was over 4.6 times slower than the macromodel simulation carried out using HSPICE, as reported in [1]. One of the reasons was that the semiconductor device models implemented in ELDO were not as efficient as those in HSPICE.

V. A VHDL-AMS behavioural fault model for the inverting opamp

A VHDL-AMS model for the behavioural model given in (1) is developed and given in Figure 11.

```
--behavioural opamp
LIBRARY DISCIPLINES;
LIBRARY IEEE;
USE
DISCIPLINES.ELECTROMAGNETIC_SYSTEM.ALL;
USE IEEE.MATH_REAL.ALL;

--entity
ENTITY op_behav IS
  GENERIC ( m : real := 0.0; --fault-free value
            k : real := 0.0; --fault-free value
           Acl : real := -1.0; --closed-loop gain
           rin : real := 4.0e5;);
 PORT (TERMINAL inn, outt : electrical);
END;

--architecture
LIBRARY DISCIPLINES;
LIBRARY IEEE;
USE
DISCIPLINES.ELECTROMAGNETIC_SYSTEM.ALL;
USE IEEE.MATH_REAL.ALL;

ARCHITECTURE behav OF op_behav IS
  quantity vout across iout through outt;
  quantity vin across iin through inn;

begin
iin  == (vin + (1.0 + m)*vin) / rin;
vout == Acl * (vin + m * vin + k);
end;
```

Figure 11. A VHDL-AMS behavioural fault model for the inverting operational amplifier for the fault-free case.

As can be seen from the Figure 11, it is much simpler to implement the behavioural model in VHDL-AMS compared with the macromodel development using SPICE-like languages.

V. Conclusion

In this paper, we have discussed behavioural and macromodelling techniques in order to speed-up analogue fault simulation process. We also have developed a behavioural fault model in VHDL-AMS for an opamp operating in inverting amplifier configuration. Capturing a circuit behaviour under faulty conditions at a higher level using mathematical equations (behavioural modelling) is somewhat simpler than trying to come up with the macromodels for that circuit. As VHDL-AMS and Verilog-AMS have now been standardised, it should be easier to develop behavioural fault models using these standard languages.

References

1. A. J. Perkins, M. Zwolinski, C. D. Chalk and B. R. Wilkins, "Fault Modeling and Simulation Using VHDL-AMS", Analog Integrated Circuits and Signal Processing, vol. 16, pp. 141-155, 1998.
2. M. Abramovici, M.A. Breuer, A.D. Friedman, "Digital Systems Testing and Testable Design", IEEE Press, 1990.
3. C.-Y. Pan and K.-T. Cheng, "Fault Macromodeling for Analog/Mixed-Signal Circuits", IEEE International Test Conference, ITC'97.
4. G. Casinovi and A. Sangiovanni-Vincentelli, "A Macromodeling Algorithm for Analog Circuits", IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 10, No. 2, pp. 150-160, February 1991.
5. G. A. Boyle, D. O. Pederson, B. M. Cohn and J. E. Solomon, "Macromodeling of Integrated Circuit Operational Amplifiers", IEEE J. of Solid State Circuits SC-9, pp. 353-363, 1974.
6. C. Chalk and M. Zwolinski, "Macromodel of CMOS operational amplifier including supply current variation", Electronics Letters 31, pp. 1398-1400, 1995.
7. P. Mandal, V. Visvanathan, "Macromodeling of the AC Characteristics of CMOS Op-Amps", IEEE 1993 Conference on Computer Aided Design, Digest of Technical Papers, pp.334-339, 1993.
8. M. E. Brinson, D. J. Faulkner, "Modular SPICE macromodel for operational amplifiers", IEE Proc.-Circuits Devices Syst., Vol. 141, No. 5, pp. 417-420, October 1994.
9. M. E. Brinson, D. J. Faulkner, "A SPICE Noise Macromodel for Operational Amplifiers", IEEE Transactions on Circuits and Systems-I: Fundamental Theory and Applications, Vol. 42, No. 3, pp. 166-168, 1995.
10. G. Krajewska and F. E. Holmes, "Macromodeling of FET/Bipolar Operational Amplifiers", IEEE Journal of Solid-State Circuits, Vol. SC-14, No. 6, pp. 1083-1087, 1979.
11. C. Turchetti and G. Masetti, "A Macromodel for Integrated All-MOS Operational Amplifiers", IEEE Journal of Solid-State Circuits, Vol. SC-18, pp. 389-394, 1983.
12. M. E. Brinson, D. J. Faulkner, "SPICE macromodel for operational amplifier power supply current sensing", Electronics Letters, Vol. 30, No. 23, 10th November 1994.
13. R. V. Peic, "Simple and Accurate Nonlinear Macromodel for Operational Amplifiers", IEEE Journal of Solid-State Circuits, Vol. 26, No. 6, pp. 896-899, 1991.
14. B. Perez et al, "A New Nonlinear Time-Domain Op-Amp Macromodel Using Threshold Functions and Digitally Controlled Network Elements", IEEE

Journal of Solid-State Circuits, Vol. 23, No. 4, pp. 959-971, 1988.
15. Y.-J. Chang et al, "A Behavior-Level Fault Model for the Closed-Loop Operational Amplifier", Journal of Information Science and Engineering, Vol. 16, No. 5, pp. 751-766, September 2000.
16. B. Al-Hashimi, "Behavioural Simulation of Filters", IEE Colloquium on Analogue simulation: the dream & the nightmare, November 1995.
17. A. I. Kayssi, K. A. Sakallah, "Macromodel Simplification Using Dimensional Analysis", 1994 Int. Symp. On Circuits and Systems, pp. 335-338, 1994.
18. M. Zwolinski, Z.R. Yang and T. J. Kazmierski, "Using robust adaptive mixing for statistical fault macromodelling", IEE Proceedings: Circuits, Devices and Systems, vol 147, no 5, pp. 265-270, Oct 2000.
19. G. Casinovi, "Multi-Level Simulation of Large Analog Systems Containing Behavioral Models", IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems, pp. 1391-1399, Vol.13, No.11, November 1994.
20. E. Bruls, et. al. , "Analogue fault simulation in standard VHDL", IEE Proc. Circuits Devices Syst., pp.380-385, Vol. 143, No. 6, December 1996.
21. C. G. Broyden, et. al., "A class of methods for solving nonlinear simultaneous equations", Mathematics of Computation, vol. 19, no. 92, pp. 577-593, 1965.
22. http://www.ee.washington.edu/mad/benchmarks/benchmarks.html
23. VHDL Language Reference Manual, IEEE Standard 1076-1993.
24. Standard Description Language Based on the Verilog™ Hardware Description Language, IEEE Standard 1364-1995.
25. http://www.ednmag.com
26. E. Christen, and K. Bakalar, "VHDL-AMS, A Hardware Description Language for Analog Applications", IEEE Transactions on Circuits and Systems-II: Analog and Digital Signal Processing, Vol.46. No.10, October 1999.
27. www.mentor.com
28. www.cadence.com
29. www.verilog.com
30. www.analogy.com
31. http://www.hamster-ams.com/
32. www.eda.org/analog

8.4.8 Peter R. Wilson, Yavuz Kılıç, J. Neil Ross, Mark Zwoliński, Andrew D. Brown, "Behavioural Modelling of Operational Amplifier Faults using analogue Hardware Description Languages", 2001 IEEE International Workshop on Behavioral Modeling and Simulation (BMAS 2001).

Behavioural Modelling of Operational Amplifier Faults using analogue Hardware Description Languages

Peter R. Wilson[1], Yavuz Kiliç [†] (Member IEEE), J. Neil Ross, Mark Zwolinski (Senior Member IEEE), Andrew D. Brown

University of Southampton,
Department of Electronics and Computer Science,
Southampton, United Kingdom,
prw99r@ecs.soton.ac.uk

[†]Philips Semiconductors,
Southampton,
United Kingdom
Yavuz.Kilic@philips.com

Abstract

The use of behavioural modelling for operational amplifiers has been well known for many years and previous work has included modelling of specific fault conditions using a macro-model. In this paper, the models are implemented in a more abstract form using analogue Hardware Description Languages (HDLs), including MAST, taking advantage of the ability to control the behaviour of the model using high-level fault condition states. The implementation method allows a range of fault conditions to be integrated without switching to a completely new model. The various transistor faults are categorised, and used to characterise the behaviour of the HDL models. Simulations compare the accuracy and speed of the transistor and behavioural level models under a set of representative fault conditions.

Keywords-Behavioural Fault Modelling, VHDL-AMS, MAST, Hardware Description languages, Simulation, Operational Amplifier

1 Introduction

It is becoming increasingly necessary to use mixed signal simulation to understand the behaviour of circuits under fault conditions. In order to practically implement simulation techniques it is necessary to implement realistic fault models, simulate large numbers of possible fault conditions in a reasonable time and test the resulting behaviour against the design specification.

If these three requirements are considered in reverse order, the first decision to make is what kind of testing approach to take? The two main types of approach are specification based and fault model based. In the specification based approach the circuit is tested against the specification [1] and a fault is deemed to have occurred only if the resulting measurement of performance is outside the specification range (e.g. rise time or bandwidth). Another approach is to build up a 'fault dictionary' of the standard faults characterised for each device or circuit [2-6]. In this case, the fault occurs when the model exhibits specific faulty behaviour that may still meet the specification. Using this approach, when the fault occurs, then the behaviour can be matched against the previously obtained fault types and immediately identified.

It is obvious from the requirements of the fault simulation approaches, especially the fault model based technique, that exhaustive simulations are required to identify the faulty behaviour. This requirement virtually mandates the intelligent use of behavioural modelling techniques to reduce the simulation times required [7]. A significant issue with the implementation and use of behavioural models in simulation for analogue integrated circuits is the matching of the device behaviour with the one found by transistor level simulation. Usually the transistor level simulation is carried out in a variety of SPICE simulators (e.g. HSPICE), and the question arises, how can the behavioural models be characterised easily and used in conjunction with these transistor level descriptions? Typical behavioural simulators, such as Saber, may use a proprietary language (MAST [8]). Despite the capability of the simulator at the behavioural level, the results of the transistor level simulations may not match those found by SPICE-like simulators, such as HPSICE, exactly (due to differences in the underlying transistor models and solution methods). The standard language IEEE 1076.1 (VHDL-AMS) [9] may also be used in simulators such as VeriasHDL or HAMSter, but the same problem arises that behavioural simulators are not renowned for their ability to deal adequately with transistor level simulations for large circuits. It is clear therefore, that checking is required at all stages to ensure that the behavioural models are consistent with the benchmark transistor level models. If this is done then minor differences between the implementations in different simulators can be minimized.

Implementing realistic fault models at different levels of abstraction is necessary to provide the required accuracy of the mixed level simulations. Using transistor level simulation models to establish the basic behaviour under a set of predefined fault conditions provides a baseline to which the behavioural model can be characterised. The choice can then be made as to which type of behavioural modelling approach is required. In this paper, the concentration is on the operational amplifiers, and there are two main approaches for behavioural modelling of these circuits; the macro-model and the equation based model. The standard Boyle macro-model [10] has been used for many years to behaviourally model opamps, but with the development of modern HDL based simulators, such as Saber, a more direct equation based implementation of

[1] This work was partly supported by a grant from the Engineering and Physical Sciences Research Council (United Kingdom) and Advanced Power Components (Rochester, United Kingdom).

opamp behaviour is possible. This paper deals with the equation-based approach to implement fault behavioural model using catastrophic and parametric structural faults.

The structure of the rest of the paper is as follows: First, transistor level opamp modelling is discussed. Then closed loop fault behavioural model is developed for a benchmark opamp circuit. Later, an open loop fault behavioural model is discussed. Finally, some conclusions are drawn.

2 Transistor Level Modelling

2.1 Introduction

While the simulation of devices at the transistor level is computationally very expensive, it is generally accepted that this provides a somewhat realistic and accurate result. It has been previously discussed in [5] and [7] how faster simulations can be carried out by using hierarchy and replacing some devices with behavioural models, without a severe penalty on the accuracy achieved. In this paper, the transistor level models are used as a benchmark to establish the behaviour of the device, and this is then used to characterise the behavioural model to the same level of accuracy, but with a much faster simulation.

2.2 Benchmark Operational Amplifier

To demonstrate the concepts used in this paper, the IEEE Mixed-Signal Benchmark opamp circuit has been used [11], the schematic for which is shown in figure 1. The SPICE netlist for this opamp is given in Figure 2, which also shows the Level 3 MOS transistor parameters used in the benchmark circuit.

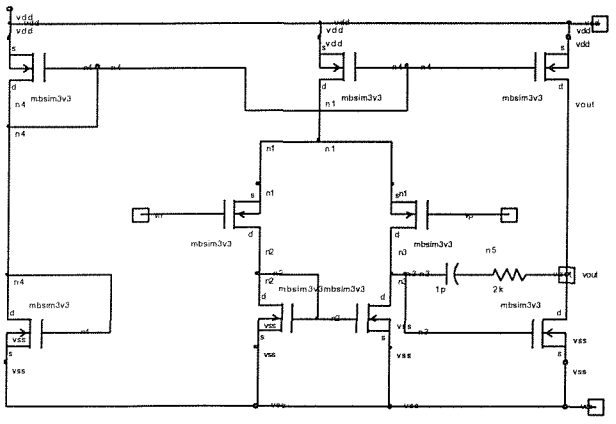


Figure 1: Transistor Level Operational Amplifier

```
.subckt OpAmpFaultFree VO VP VN Vdd Vss
Rc NET32 VO 2E3 M=1.0
Cc NET48 NET32 1E-12 M=1.0
M6 NET48 VP NET44 Vdd PMOS L=4E-6 W=30E-6 M=1.0
M3 NET35 VN NET44 Vdd PMOS L=4E-6 W=30E-6 M=1.0
M9 VO NET48 Vss Vss NMOS L=3E-6 W=154.2E-6 M=1.0
M4 NET35 NET35 Vss Vss NMOS L=4E-6 W=15E-6 M=1.0
M7 NET48 NET35 Vss Vss NMOS L=4E-6 W=15E-6 M=1.0
M2 NET54 NET54 Vss Vss NMOS L=32E-6 W=3E-6 M=1.0
M8 VO NET54 Vdd Vdd PMOS L=4E-6 W=200E-6 M=1.0
M1 NET54 NET54 Vdd Vdd PMOS L=4E-6 W=12E-6 M=1.0
M5 NET44 NET54 Vdd Vdd PMOS L=4E-6 W=30E-6 M=1.0
.MODEL NMOS NMOS LMIN=1.2E-06 LMAX=1.5E-06
WMIN=2.0E-06 WMAX=500E-06 LEVEL=3
+VTO=.79 GAMMA=.38 PHI=.53 RD=63 RS=63 IS=1E-16
PB=.8 CGSO=1.973E-10
+CGDO=1.973E-10 RSH=45 CJ=0.00029 MJ=.486
CJSW=3.3E-10 MJSW=.33 JS=0.0001
+TOX=2.5E-08 NSUB=8.7E+15 NFS=8.2E+11 TPG=1 XJ=1E-
07 LD=7E-08 UO=577
+VMAX=150000 FC=.5 DELTA=.3551 THETA=0.046 ETA=.16
KAPPA=0.05
.MODEL PMOS PMOS LMIN=1.2E-06 LMAX=100E-06
WMIN=2.0E-06 WMAX=500E-06 LEVEL=3
+VTO=-8.40000000E-01 GAMMA=.53 PHI=.58 RD=94 RS=94
IS=1E-16 PB=.8
+CGSO=3.284E-10 CGDO=3.284E-10 RSH=100 CJ=0.00041
MJ=.54 CJSW=3.4E-10 MJSW=.3
+JS=0.0001 TOX=2.5E-08 NSUB=1.75E+16 NFS=8.4E+11
TPG=1 XJ=0 LD=6E-08 UO=205
+VMAX=500000 FC=.5 DELTA=.4598 THETA=.14 ETA=.17
KAPPA=10
.ends OpAmpFaultFree
```

Figure 2: Transistor Level Operational Amplifier Netlist

2.3 Transistor Level Fault Free Behaviour

Using the transistor level operational amplifier model described, the behaviour of the device from the inputs to the output can be characterised in the inverting, non-inverting and unity gain configurations using the test circuits given in figures 3(a), (b), and (c), respectively.

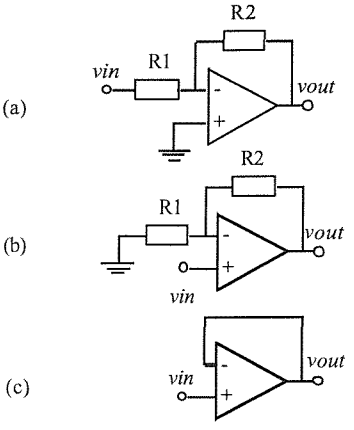


Figure 3: Closed Loop Test Circuits (a) Inverting, (b) Non-inverting, (c) Unity Gain

Using these test circuits, the fault free operational amplifier model was tested using Hspice and Pspice, with the input offset voltage and the output voltage measured in each case. The results of these simulations are shown in figures 4, 5, and, 6. The DC transfer analysis was used in this case as the faults could be classified using this aspect of the device behaviour alone. The DC transfer analysis in Saber allows the specification of the starting and finishing voltage values (-3V and +3V respectively), and a voltage step size (10mV in this study).

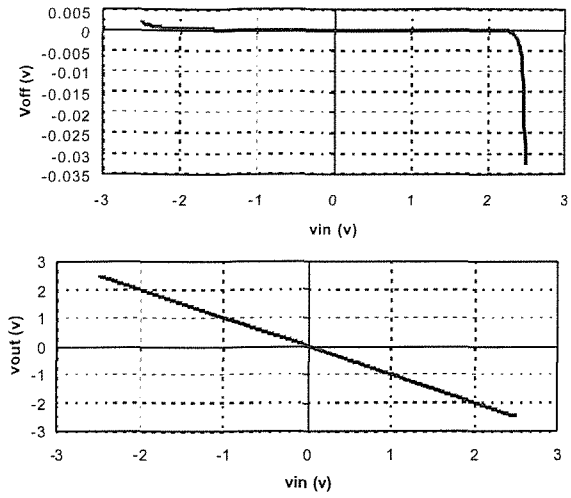


Figure 4: Inverting Amplifier Input Offset Voltage (vp-vn) and Output Voltage

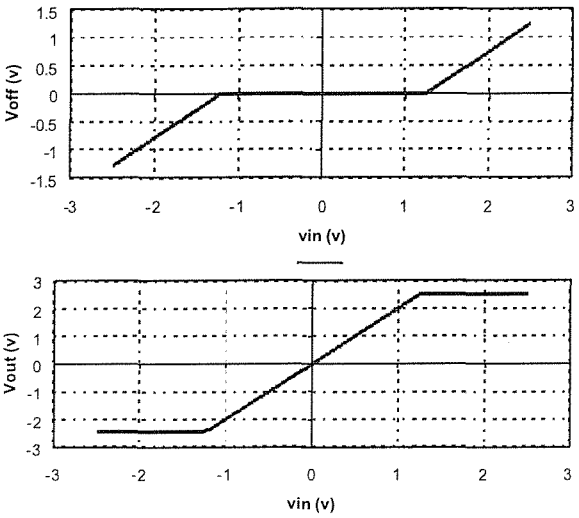


Figure 5: Non-Inverting Amplifier Input Offset Voltage (vp-vn) and Output Voltage

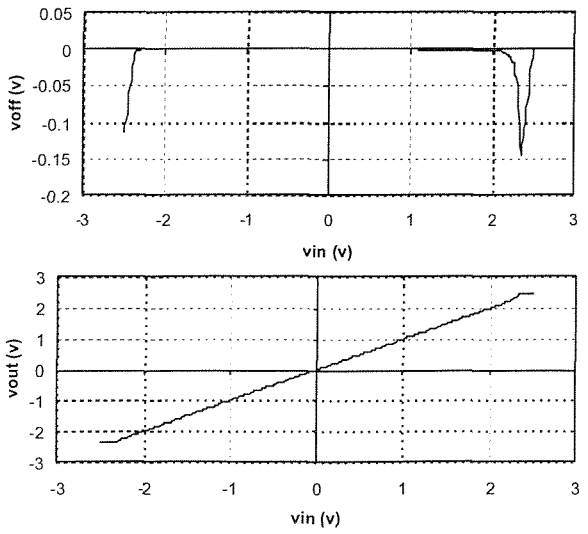


Figure 6: Unity Gain Amplifier Input Offset Voltage (vp-vn) and Output Voltage

2.4 Operational Amplifier Fault Behaviour

Recently work has been done to group the catastrophic and parametric faults that can occur in operational amplifiers by looking at the offset voltage at the inputs of the opamp, while carrying out DC sweep analysis [11]. Catastrophic faults are those that occur when an open or short circuit causes a complete failure in the operation of the device. Parametric faults, on the other hand, are variations in the MOS transistor channel lengths and widths, and threshold voltages, which cause a minor variation in the device's specification, such as gain and bandwidth. In this paper we are concentrating on the catastrophic faults, but the same models can be used to characterise the parametric faults as well.

The catastrophic faults for the opamp shown in Figure 1 can be categorised into four main types, type I (M5 Drain-Gate Short), type II (M7 Drain open), type III (M5 Drain to Source Short) and type IV (M5 Drain-Gate Short). Fault types I-III are for the inverting amplifier configuration, while the type IV fault applies to the non-inverting configuration. Figures 7-10 show DC transfer characteristics for four fault types.

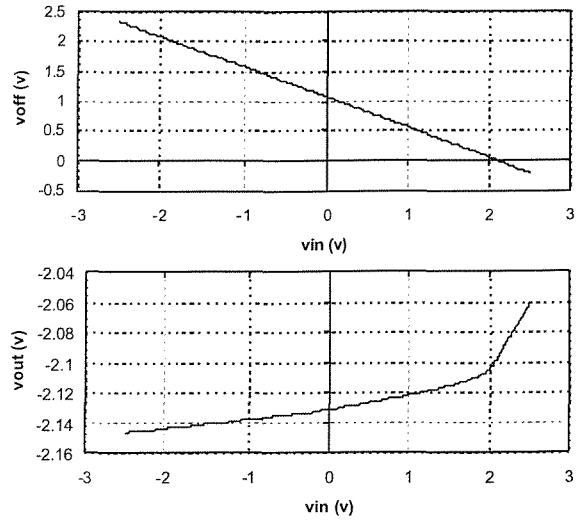


Figure 7: Inverting Amplifier Input Offset Voltage (vp-vn) and Output Voltage with type I fault

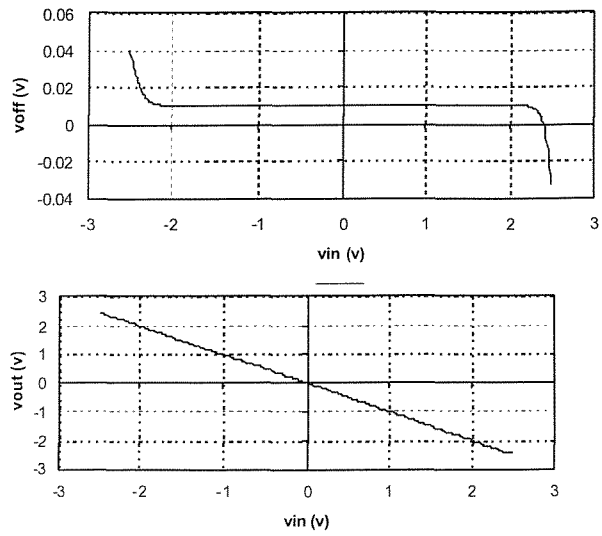


Figure 8: Inverting Amplifier Input Offset Voltage (vp-vn) and Output Voltage with type II fault

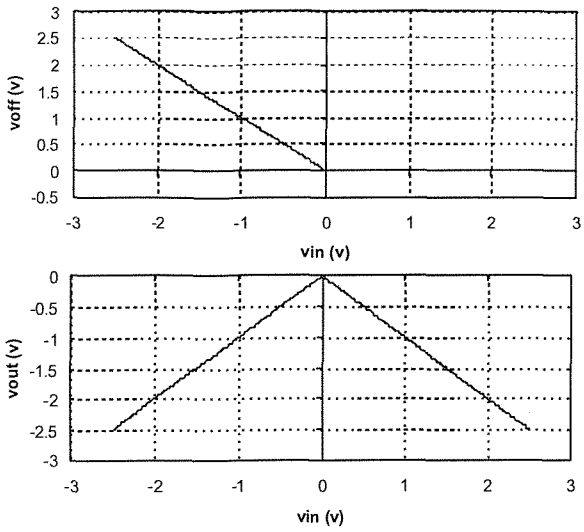


Figure 9: Inverting Amplifier Input Offset Voltage (vp-vn) and Output Voltage with type III fault

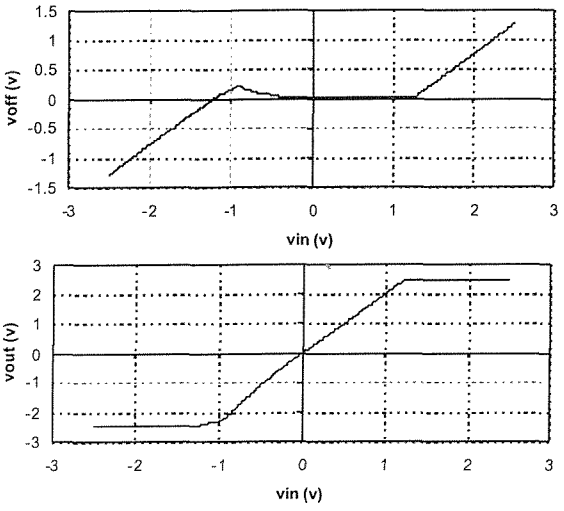


Figure 10: Non-Inverting Amplifier Input Offset Voltage (vp-vn) and Output Voltage with type IV fault

3 Closed Loop Behavioural Modelling

3.1 Close Loop Model Equation

With the complete transistor level modelling of the operational amplifier for the fault free and faults I-IV completed, this behaviour can be modelled behaviourally in a closed loop form. Change et al [11] provide a simple closed loop model of the form given in (1) which gives the input-output voltage relationship of the behavioural model.

$$V_{out} = A_{CL}\left[(1+m)V_{in} + k\right] \qquad (1)$$

where A_{CL} is the Closed Loop gain of the opamp, and m and k are parameters that characterise the non-ideal opamp effects, such as the limited output resistance, and the opamp's faulty behaviour for the closed loop configuration.

How the parameters m and k can be derived analytically from the parameters of the opamp is described in [7] and [11]. In this paper, the parameters were derived directly using the transistor level simulation results. For the fault free case, the parameters can be derived by inspection. For example, in the fault free inverting case, the gain is –1 (R_1 = R_2 = 1MegΩ), and therefore m=0 and k=0. For type I faults (a stuck at fault), if m=-1, then the value of the output voltage will simply be k, where k is the magnitude of the stuck at voltage.

3.2 Proposed Closed-Loop Fault Behavioural Model

The model defined by equation (1) was implemented as a behavioural model using the MAST modelling language and simulated with the Saber simulator. The model listing is provided in Figure 11.

```
template opamp_behav2 vin voutgnd = a,m,k
electrical vin,vout,gnd
#...Operational Amplifier Parameters
number a=1
#...Fault Offset Voltage Parameters
number m=0
number k=0
{
#...Declarations
var i i
val v vo,vi,fo,voutcalc
#...Procedural Expressions
values {
        #...Terminal Voltages
        vo = v(vout) - v(gnd)
        vi = v(vin) - v(gnd)
        #...Fault Offset Voltage
        fo = m*vi + k
        voutcalc = a*(vi+fo)
        #...Supply Voltage Limit
        if (voutcalc > 2.5) voutcalc=2.5
        if (voutcalc< -2.5) voutcalc = -2.5
}
equations {
        #...Fundamental Equations
        i(vout->gnd) += i
        vo = voutcalc
        }
}
```

Figure 11: Closed Loop opamp MAST model

3.3 Testing the basic fault model

Using the same test benches as were used in the transistor level simulations, the behavioural model was tested in the fault free and each of the fault type cases with a DC transfer analysis. In the inverting fault free case, m=k=0 with the output voltage as shown in figure 12.

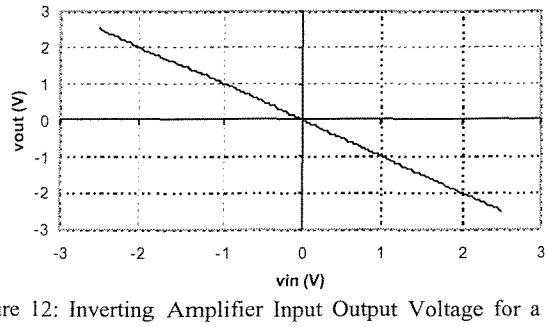


Figure 12: Inverting Amplifier Input Output Voltage for a fault free behavioural model

In the fault I case, the output voltage is a stuck at voltage, with the values -2.14 at -2.5V input to -2.11 at +2V. The region 2V to 2.5V is slightly steeper, with the variation -2.11 to -2.06 for the input range 2V to 2.5V. Therefore to

get a highly accurate mapping of output voltage behaviourally would require a PWL model. However, in this case the fact that the model is exhibiting a stuck at fault is adequate, and is accurate to within 5%. To get equation (1) to exhibit this behaviour, m=-1 (cancelling out the input voltage terms) and, k is just the value of stuck at voltage, which is probably best matched at vin=0, $vout$=-2.13V, therefore k=2.13. The Saber simulation result of the behavioural model for this fault type is given in Figure 13.

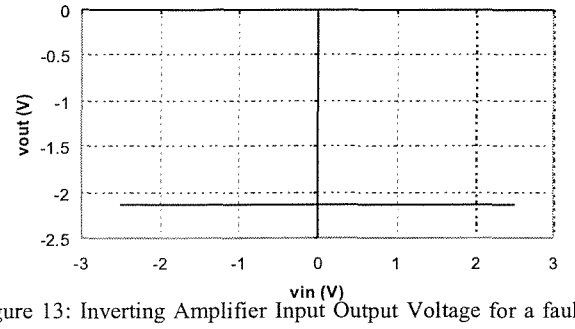


Figure 13: Inverting Amplifier Input Output Voltage for a fault I behavioural model

In the type II faults, the opamp works almost correctly, but the input offset is much higher than normal (10mV), causing a slight offset in the output. This manifests itself with a slight offset in the output voltage, leading to early saturation on one side of the output voltage swing. The basic behavioural model does not cope with this and as such must in fact include a limiting function to limit the output voltage to the supply rails (+/- 2.5V in this case). The resulting simulation of the output voltage is shown in figure 14.

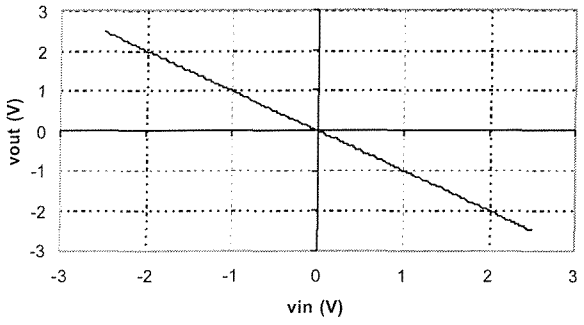


Figure 14: Inverting Amplifier Input Output Voltage for a fault II behavioural model

In the type III fault case, when the input voltage is greater than zero, the inverting opamp circuit works correctly, but when vin<0, the circuit behaviour turns into non-inverting. In terms of the equation (1), this implies that k=0 and that m is 0 for vin>0 and for vin<0, m=-2. This is summarised in equation (2).

$$vout = \begin{cases} A_{cl} vin & vin > 0 \\ A_{cl}(vin + m * vin) & vin < 0, m = -2 \end{cases} \qquad (2)$$

This discontinuous behaviour cannot be modelled using the simple behavioural model previously given, and a modification was made in the model to include this change in behaviour as shown in Figure 15.

```
#...Fault Offset Voltage
if (vi <0) {
        fo = -2*vi
}
else {
        fo = 0
}
vout = a*(vi+fo)
```

Figure 15: Modification to the behavioural model for type III faults

Using this modified model, the resulting behaviour can be seen in figure 16.

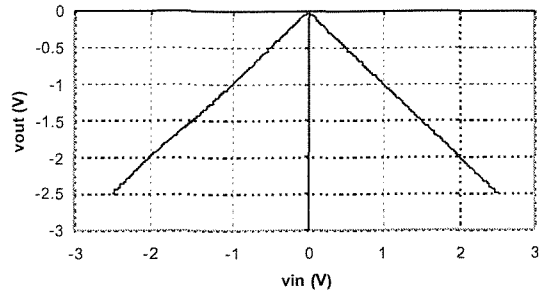


Figure 16: Inverting Amplifier Input Output Voltage for a fault III behavioural model

The same behavioural model can also be used for the non-inverting closed-loop case. In this configuration, the transistor level behaviour can be replicated using the parameters m=0 and k=0 with A_{CL}=2 ($A_{CL} = 1 + R_2 / R_1$ for the non-inverting opamp). The resulting simulated output voltage is shown in Figure 17.

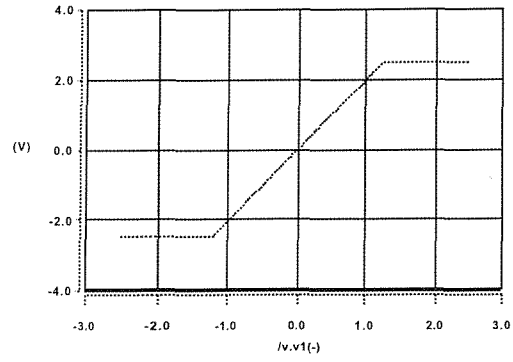


Figure 17: Non-Inverting Amplifier Input Output Voltage for a fault free behavioural model

On inspection of the transistor level results for the fault IV case, it is clear that a PWL approximation is required to provide a realistic match with a behavioural model. The closed loop model was therefore modified using a PWL voltage offset as shown in Figure 18.

```
#...Definition of the PWL structure
struc {
        number vi, vo
        }   pwlv[*]   =   [(-2.15,0.016),(-
1.3,0.02),(-1.13,-
0.111),(0.6,0.003),(2.5,0)]
... Existing Model Code
#... Calculation of the fault voltage
vos = pwl1(2,pwlv,vin)
```

Figure 18: Modification to the behavioural model for type IV faults

The resulting change in behaviour is subtle, but is a slight non-linearity introduced on the output voltage. The output voltage obtained using the behavioural simulation is given in Figure 19.

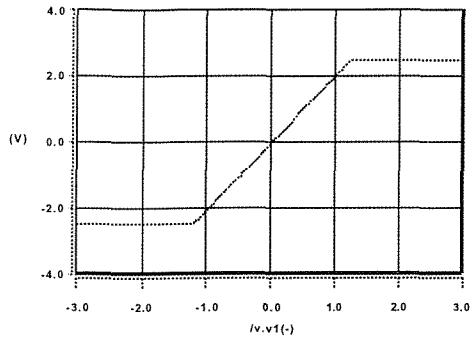


Figure 19: Non-Inverting Amplifier Input Output Voltage for a fault IV behavioural model

4 Open Loop Behavioural Modelling

One drawback with the closed loop model is the restriction on the topologies that can be simulated. If extra components are added into the feedback loop for example, then a complete re-characterisation is needed. To combat this, therefore, a modified fault offset voltage model was created that could be connected to the input of the opamp creating the required fault offset voltage, while allowing an arbirtrary connection to the opamp externally. The resulting mast model is given in figure 20.

```
template fos inp inm fosp=m,k,fault
electrical inp,inm,fosp
number m=0,k=0
enum { _fault1, _fault2, _fault3, _fault4, _none,
_specified } fault=_none
{
var i i
val v vin,vos
foreign pwl1

struc {
        number vi, vo
        } pwlv[*] = [(-2.15,0.016),(-1.3,0.02),(-
1.13,-0.111),(0.6,0.003),(2.5,0)]
#       }       pwlv[*]     =     [(-2.5,0),(-1.2,0),(-
0.9,0.2),(-0.2,0.01),(1.0,0),(2.5,0)]

#...Procedural Statements in this section
values {
        #...Calculate   input   voltage   vin   from
voltage on pins inp and inm
        vin = v(inp)- v(inm)
        if (fault == _fault1) {
                vos=-1.02*vin + -2.215
        }
        else if (fault == _fault2) {
                vos = -0.011
        }
        else if (fault == _fault3) {
                if (vin < 0 ) {
                        vos = 0
                }
                else {
                        vos = -2*vin
                }
        }
        else if (fault == _fault4 ) {
                vos = pwl1(2,pwlv,vin)
                vos = -0.5*vos
        }
        else if (fault == _specified) {
                vos = vin*m + k
        }
        else {
                vos = 0
        }
}

equations {
        i(fosp->inp) += i
        i : v(inp) - v(fosp) = vos

        }
}
```

Figure 20: General Purpose Opamp Fault Model

This model can then be connected in series with any behavioural opamp model with the option of either specifying specific fault types, or defining the *m* & *k* parameters directly. The advantage of using this type of approach becomes clear when more complex circuits are tested such as the IEEE Mixed-Signal Benchmark Biquad filter shown in Figure 21. Obviously, the opamps are not the simple buffers previously analysed using the closed loop model, and as such the open loop fault model becomes an ideal approach to simulating faults in this type of circuit. If this circuit is simulated in the fault free case the resulting output signal is given in Figure 22.

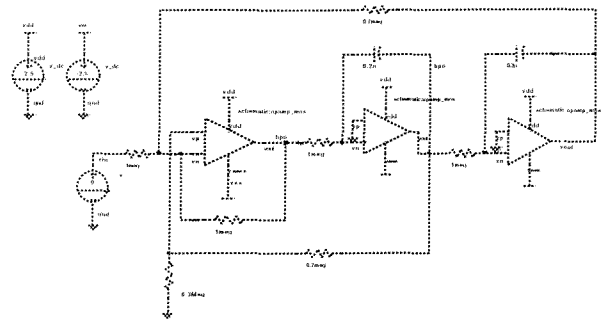


Figure 21: Biquad Filter Benchmark circuit

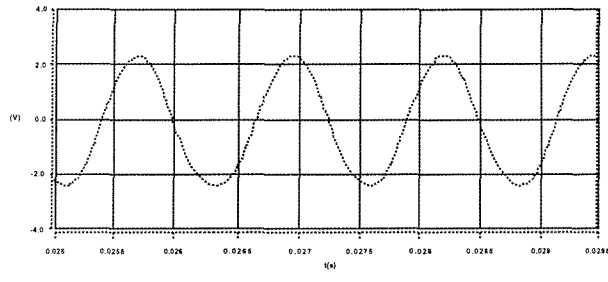


Figure 22: Biquad Filter fault free transient response

If the fault model for a type I fault is implemented in any of the opamps using the open loop behavioural model, then the output will be a stuck at voltage, as is shown in Figure 23.

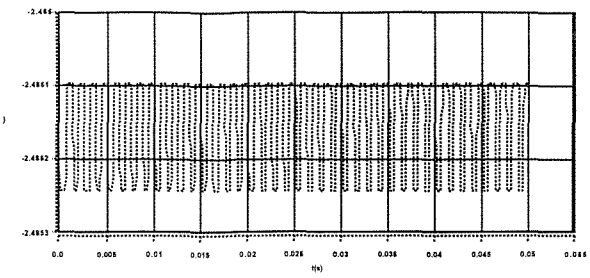


Figure 23: Biquad Filter fault I transient response

There is a slight ripple on the output, but essentially the output is stuck at -2.4V.

5 Summary of the results

It is clear from these results that there is a good correlation between the transistor level and behavioural level models. The real benefit for multiple simulations depends also on the simulation times in each case, and these are summarised in table 1. For each case a number of runs (3) was taken in each case and the average simulation time (CPU seconds) was recorded.

Circuit configuration	Transistor level Simulation Time (s)	Behavioural level Simulation Time (s)
Inverting Fault Free	1.88	0.1
Inverting Fault I	1.78	0.1
Inverting Fault II	1.81	0.1
Inverting Fault III	1.89	0.15
Non-Inverting Fault Free	1.89	0.1
Non-Inverting Fault IV	1.89	0.45
Biquad Filter (a)	11.1	2.35
Biquad Filter (b)	93.2	2.49

Table 1 : Comparison of Simulation Times[2]

In most of the opamp test cases, the speed up in using behavioural models was 18 times. This was slightly reduced in the Fault III and Fault IV models due to the extra solution time required to deal with the more complex PWL characteristic in the model. For the Biquad filter in case (a) the input voltage was half the supply voltage, and in case (b) the input was full scale. It is clear from the difference in simulation times that the relative merits of the MOS and Behavioural models may depend significantly on the operating region of the devices. If the devices are pushed into their non-linear regions, then the solution time may drastically increase as is the case here (x9). It is interesting to note that the behavioural model simulation time is almost exactly the same in both cases implying that the model is robust and can handle the limiting to the supply aspects without undue convergence difficulties.

6 Conclusions

In this paper, a method of implementing fault behavioural models for operational amplifiers has been presented. Previous work has been extended to cover both the open and closed loop configurations allowing greater flexibility in the application of the fault models in the general case. Results show a good correlation between transistor and behavioural models at all stages, with a corresponding improvement in simulation times.

[2] All the simulation times in table 1 were obtained using Saber 5.1 running on a Celeron 500MHz PC with Windows NT.

7 References

[1] Voorakaranam R. and Chatterjee A., "Hierarchical specification driven analogue fault modeling for efficient fault simulation and diagnosis", *IEEE International Test Conference*, 1997, pp903-912

[2] Perkins A.J., Zwolinski M.J., Chalk C.D. & Wilkins B.R., "Fault Modeling and Simulation using VHDL-AMS", *Analogue Integrated Circuits and Signal Processing*, Vol. 16, 1998, pp141-155

[3] Pan C.Y. & Cheng K.T., "Fault Macromodeling for Analogue/Mixed Signal Circuits", *IEEE International Test Conference*, 1997

[4] Milor L. & Visvanathan V., "Detection of catastrophic faults in analogue integrated circuits", *IEEE Transactions on Computer Aided Design (CAD)*, Vol. 8, No. 2, 1989, pp114-130

[5] Sounder T.M. & Stenbakken G.N., "A comprehensive approach for modeling and testing analogue and mixed signal devices", *Proceedings of International Test Conference*, 1990, pp169-176

[6] Nagi N., Chatterjee A., Balivida A., & Abraham J.A., "Fault based automatic test generator for linear analogue circuits", *Proceedings of International Conference on CAD*, 1993, pp88-91

[7] Kilic Y. & Zwolinski M., "Behavioural/Macro modelling to speed up analogueue fault simulation", ELECO 2001 (to be presented)

[8] MAST Reference Manual, Avant! Corporation , USA

[9] IEEE Standard 1076.1 Analogue and Mixed Signal Extensions to VHDL

[10] Boyle G.A., Pederson D.O., Cohn B.M. & Solomon J.E., "Macromodeling of Integrated Circuit Operational Amplifiers", *IEEE Journal of Solid State Circuits*, 1974, pp353-363

[11] Chang Y., Lee C.L., Chen J.E., & SU C., "A Behaviour Level Fault Model for the Closed-Loop Operational Amplifier", *Journal of Information Science and Engineering*, 2000, Vol. 16, pp751-766

8.4.9 Peter R. Wilson, Yavuz Kılıç, J. Neil Ross, Mark Zwoliński, Andrew D. Brown, "Behavioural Modelling of Operational Amplifier Faults using VHDL-AMS", Design, Automation and Test in Europe (DATE) Conference & Exhibition 2002, Paris, France, 4-8 March 2002, Le Palais des Congrès.

Behavioural Modelling of Operational Amplifier Faults using VHDL-AMS

Abstract

The use of behavioural modelling for operational amplifiers has been well known for many years and previous work has included modelling of specific fault conditions using a macro-model. In this paper, the models are implemented in a more abstract form using an Analogue Hardware Description Language (AHDL), VHDL-AMS, taking advantage of the ability to control the behaviour of the model using high-level fault condition states. The implementation method allows a range of fault conditions to be integrated without switching to a completely new model. The various transistor faults are categorised, and used to characterise the behaviour of the HDL models. Simulations compare the accuracy and speed of the transistor and behavioural level models under a set of representative fault conditions.

Keywords-Behavioural Fault Modelling, VHDL-AMS, Hardware Description languages, Simulation, Operational Amplifier

1 Introduction

It is becoming increasingly necessary to use mixed signal simulation to understand the behaviour of circuits under fault conditions. In order to practically implement simulation techniques it is necessary to implement realistic fault models, simulate large numbers of possible fault conditions in a reasonable time and test the resulting behaviour against the design specification.

If these three requirements are considered in reverse order, the first decision to make is what kind of testing approach to take? The two main types of approach are specification based and fault model based. In the specification based approach the circuit is tested against the specification [1] and a fault is deemed to have occurred only if the resulting measurement of performance is outside the specification range (e.g. rise time or bandwidth). Another approach is to build up a 'fault dictionary' of the standard faults characterised for each device or circuit [2-6]. In this case, the fault occurs when the model exhibits specific faulty behaviour that may still meet the specification. Using this approach, when the fault occurs, then the behaviour can be matched against the previously obtained fault types and immediately identified.

It is obvious from the requirements of the fault simulation approaches, especially the fault model based technique, that exhaustive simulations are required to identify the faulty behaviour. This requirement virtually mandates the intelligent use of behavioural modelling techniques to reduce the simulation times required [7]. A significant issue with the implementation and use of behavioural models in simulation for analogue integrated circuits is the matching of the device behaviour with the one found by transistor level simulation. Usually the transistor level simulation is carried out in a variety of SPICE simulators (e.g. HSPICE), and the question arises, how can the behavioural models be characterised easily and used in conjunction with these transistor level descriptions? Typical behavioural simulators, such as Saber, may use a proprietary language (MAST [8]). Despite the capability of the simulator at the behavioural level, the results of the transistor level simulations may not match those found by SPICE-like simulators, such as HPSICE, exactly (due to differences in the underlying transistor models and solution methods). The standard language IEEE 1076.1 (VHDL-AMS) [9] may also be used in simulators such as VeriasHDL [13] or HAMSter [14], but the same problem arises that behavioural simulators are not renowned for their ability to deal adequately with transistor level simulations for large circuits. It is, therefore, clear that checking is required at all stages to ensure that the behavioural models are consistent with the transistor level models. If this is done then minor differences between the implementations in different simulators can be minimized.

Implementing realistic fault models at different levels of abstraction is necessary to provide the required accuracy of the mixed level simulations. Using transistor level simulation models to establish the basic behaviour under a set of predefined fault conditions provides a baseline to which the behavioural model can be characterised. The choice can then be made as to which type of behavioural modelling approach is required. In this paper, the concentration is on the operational amplifiers, and there are two main approaches for behavioural modelling of these circuits; the macro-model and the equation based model. The standard Boyle macro-model [10] has been used for many years to behaviourally model opamps in the macro-model approach, but with the development of modern HDL based simulators, such as Saber, a more direct equation based implementation of opamp behaviour is possible. This paper deals with the equation-based approach to implement fault behavioural model using catastrophic and parametric structural faults, building on previous work [12] to extend the behavioural model implementation to VHDL-AMS.

The structure of the rest of the paper is as follows: First, transistor level opamp modelling is discussed. Then closed loop fault behavioural model is developed for a benchmark opamp circuit. Later, an open loop fault behavioural model is discussed. Finally, some conclusions are drawn.

2 Transistor Level Modelling

2.1 Introduction

While the simulation of devices at the transistor level is computationally very expensive, it is generally accepted that this provides a somewhat realistic and accurate result. It has been previously discussed in [5] and [7] how faster simulations can be carried out by using hierarchy and replacing some devices with behavioural models, without a severe penalty on the accuracy achieved. In this paper, the transistor level models are used as a benchmark to establish the behaviour of the device, and this is then used to characterise the behavioural model to the same level of accuracy, but with a much faster simulation.

2.2 Benchmark Operational Amplifier

To demonstrate the concepts used in this paper, the IEEE Mixed-Signal Benchmark opamp circuit has been used [11], the schematic for which is shown in figure 1. The SPICE netlist for this opamp is given in Figure 2, which also shows

the Level 3 MOS transistor parameters used in the benchmark circuit.

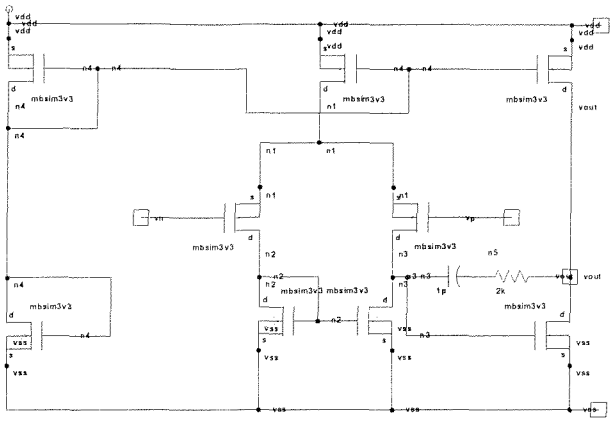


Figure 1: Transistor Level Operational Amplifier

```
.subckt OpAmpFaultFree VO VP VN Vdd Vss
Rc NET32 VO 2E3 M=1.0
Cc NET48 NET32 1E-12 M=1.0
M6 NET48 VP NET44 Vdd PMOS L=4E-6 W=30E-6 M=1.0
M3 NET35 VN NET44 Vdd PMOS L=4E-6 W=30E-6 M=1.0
M9 VO NET48 Vss Vss NMOS L=3E-6 W=154.2E-6 M=1.0
M4 NET35 NET35 Vss Vss NMOS L=4E-6 W=15E-6 M=1.0
M7 NET48 NET35 Vss Vss NMOS L=4E-6 W=15E-6 M=1.0
M2 NET54 NET54 Vss Vss NMOS L=32E-6 W=3E-6 M=1.0
M8 VO NET54 Vdd Vdd PMOS L=4E-6 W=200E-6 M=1.0
M1 NET54 NET54 Vdd Vdd PMOS L=4E-6 W=12E-6 M=1.0
M5 NET44 NET54 Vdd Vdd PMOS L=4E-6 W=30E-6 M=1.0
.MODEL NMOS NMOS LMIN=1.2E-06 LMAX=1.5E-06
WMIN=2.0E-06 WMAX=500E-06 LEVEL=3
+VTO=.79 GAMMA=.38 PHI=.53 RD=63 RS=63 IS=1E-16
PB=.8 CGSO=1.973E-10
+CGDO=1.973E-10 RSH=45 CJ=0.00029 MJ=.486
CJSW=3.3E-10 MJSW=.33 JS=0.0001
+TOX=2.5E-08 NSUB=8.7E+15 NFS=8.2E+11 TPG=1 XJ=1E-
07 LD=7E-08 UO=577
+VMAX=150000 FC=.5 DELTA=.3551 THETA=0.046 ETA=.16
KAPPA=0.05
.MODEL PMOS PMOS LMIN=1.2E-06 LMAX=100E-06
WMIN=2.0E-06 WMAX=500E-06 LEVEL=3
+VTO=-8.40000000E-01 GAMMA=.53 PHI=.58 RD=94 RS=94
IS=1E-16 PB=.8
+CGSO=3.284E-10 CGDO=3.284E-10 RSH=100 CJ=0.00041
MJ=.54 CJSW=3.4E-10 MJSW=.3
+JS=0.0001 TOX=2.5E-08 NSUB=1.75E+16 NFS=8.4E+11
TPG=1 XJ=0 LD=6E-08 UO=205
+VMAX=500000 FC=.5 DELTA=.4598 THETA=.14 ETA=.17
KAPPA=10
.ends OpAmpFaultFree
```

Figure 2: Transistor Level Operational Amplifier Netlist

2.3 Transistor Level Fault Free Behaviour

Using the transistor level operational amplifier model described, the behaviour of the device from the inputs to the output can be characterised in the inverting, non-inverting and unity gain configurations using the test circuits given in figures 3(a), (b), and (c), respectively.

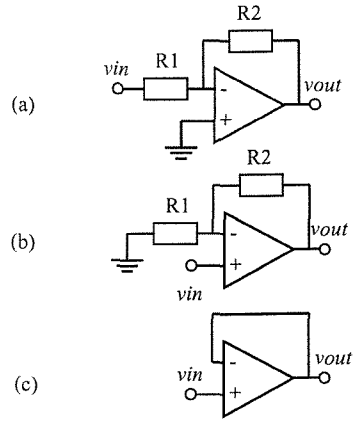


Figure 3: Closed Loop Test Circuits (a) Inverting, (b) Non-inverting, (c) Unity Gain

Using these test circuits, the fault free operational amplifier model was tested using Hspice, Pspice, and Saber with the input offset voltage and the output voltage measured in each case. The results of simulations using Saber are shown in figures 4, 5, and, 6. The DC transfer analysis was used in this case as the faults could be classified using this aspect of the device behaviour alone. The DC transfer analysis in Saber allows the specification of the starting and finishing voltage values (-3V and +3V respectively), and a voltage step size (10mV in this study).

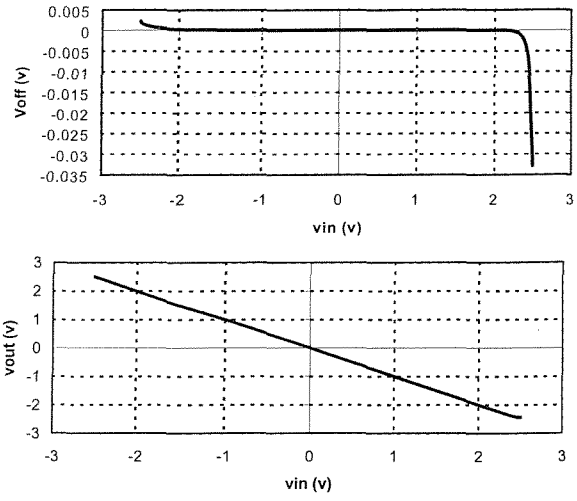
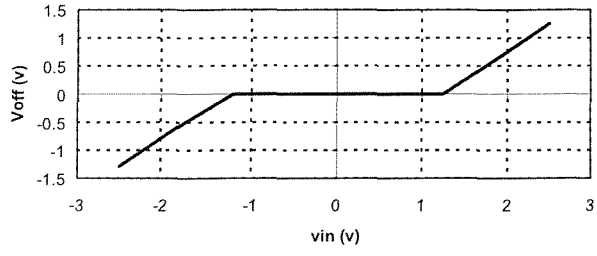


Figure 4: Inverting Amplifier Input Offset Voltage (vp-vn) and Output Voltage



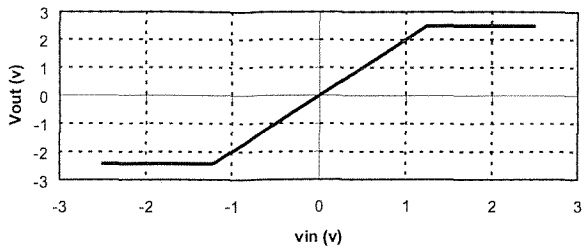


Figure 5: Non-Inverting Amplifier Input Offset Voltage (vp-vn) and Output Voltage

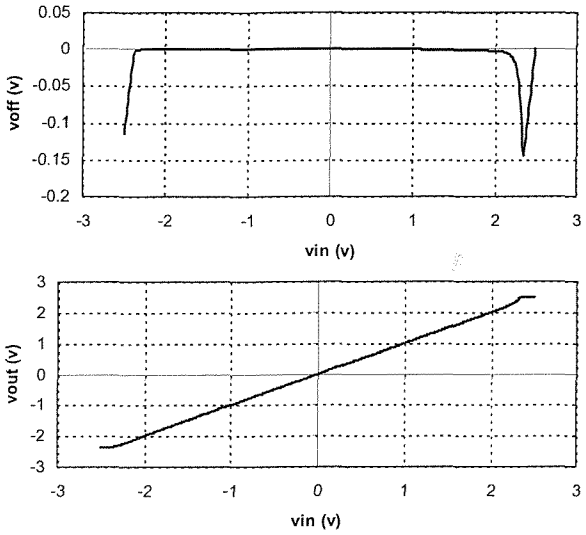


Figure 6: Unity Gain Amplifier Input Offset Voltage (vp-vn) and Output Voltage

2.4 Operational Amplifier Fault Behaviour

Recently work has been done to group the catastrophic and parametric faults that can occur in operational amplifiers by looking at the offset voltage at the inputs of the opamp, while carrying out DC sweep analysis [11]. Catastrophic faults are those that occur when an open or short circuit causes a complete failure in the operation of the device. Parametric faults, on the other hand, are variations in the MOS transistor channel lengths and widths, and threshold voltages, which cause a minor variation in the device's specification, such as gain and bandwidth. Catastrophic faults are due to local spot defects whereas parametric faults are usually due to global defects on the silicon wafer in the manufacturing process [4]. Note that some open or short circuit faults could also cause only parametric variations in the behaviour of the circuit, such as type IV faults. In this paper we are concentrating on the open and short circuit faults.

The open and short circuit faults for the opamp shown in Figure 1 can be categorised into four main types [11], type I (M5 Drain-Gate Short), type II (M7 Drain open), type III (M5 Drain to Source Short) and type IV (M5 Drain-Gate Short). Fault types I-III are for the inverting amplifier configuration, while the type IV fault applies to the non-inverting configuration. Figures 7-10 show DC transfer characteristics for four fault types, simulated using Saber.

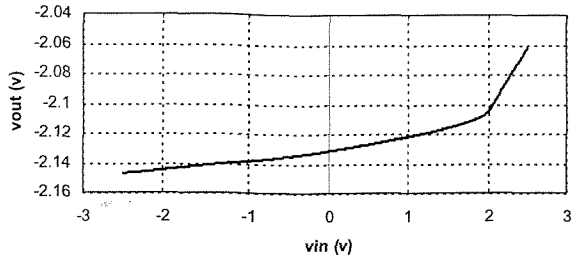


Figure 7: Inverting Amplifier Output Voltage with type I fault

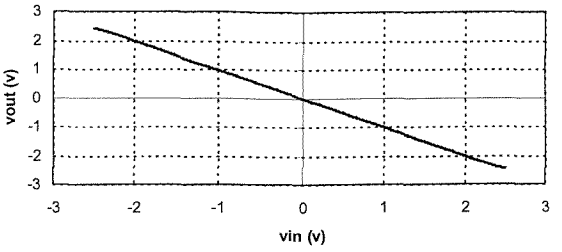


Figure 8: Inverting Amplifier Output Voltage with type II fault

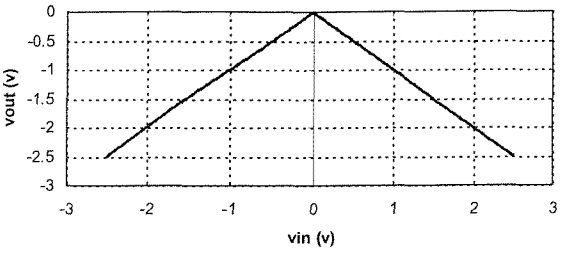


Figure 9: Inverting Amplifier Output Voltage with type III fault

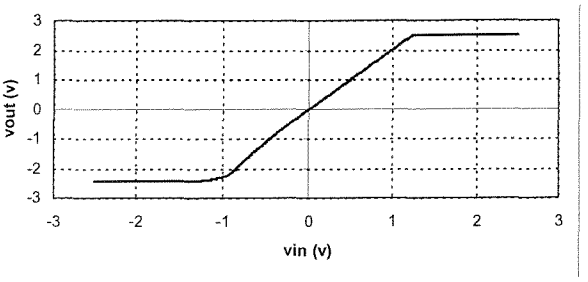


Figure 10: Non-Inverting Amplifier Output Voltage - type IV fault

3 Closed Loop Behavioural Modelling

3.1 Close Loop Model Equation

With the complete transistor level modelling of the operational amplifier for the fault free and faults I-IV completed, this behaviour can be modelled behaviourally in a closed loop form. Change et al [11] provide a simple closed loop model of the form given in (1) which gives the input-output voltage relationship of the behavioural model.

$$V_{out} = A_{CL}\left[(1+m)V_{in} + k\right] \qquad (1)$$

where A_{CL} is the Closed Loop gain of the opamp, and m and k are parameters that characterise the non-ideal opamp effects, such as the limited output resistance, and the opamp's faulty behaviour for the closed loop configuration.

How the parameters m and k can be derived analytically from the parameters of the opamp is described in [7] and [11]. In this paper, the parameters were derived directly using the transistor level simulation results. For the fault free case, the parameters can be derived by inspection. For example, in the fault free inverting case, the gain is -1 ($R_1 = R_2 = 1\text{Meg}\Omega$),

and therefore m=0 and k=0. For type I faults (a stuck at fault), if m=-1, then the value of the output voltage will simply be -k, where k is the magnitude of the stuck-at voltage.

3.2 Proposed Closed-Loop Behavioural Model

The model defined by equation (1) was implemented as a behavioural model using VHDL-AMS with the model listing provided in Figure 11.

```
entity op_behav is
   generic ( m : real := 0.0;
             k : real := 0.0;
             Acl : real := -1.0;
             rin : real := 100.0e6);
       port (terminal inn,outn: electrical);
end;

architecture behav of op_behav is
   quantity vout across iout through outn;
   quantity vin across iin through inn;
   quantity Fos : real;
   constant v_limit : real := 2.5;
begin
procedural is
  variable vout_calc : real;
  begin
   Fos := m*vin + k;
   vout_calc := Acl * (vin + Fos);
   iin  := (vin - Fos) / rin;
   if (vout_calc > v_limit) then
       vout := 2.5;
   elsif (vout_calc < -v_limit) then
       vout := -2.5;
   else vout := vout_calc;
   end if;
end procedural;
end;
```

Figure 11: Closed Loop opamp VHDL-AMS model

3.3 Testing the basic fault model

Using the same test-benches as were used in the transistor level simulations, the behavioural model was tested in the fault free and each of the fault type cases with a DC transfer analysis. In the inverting fault free case, m=k=0 with the output voltage as shown in figure 12.

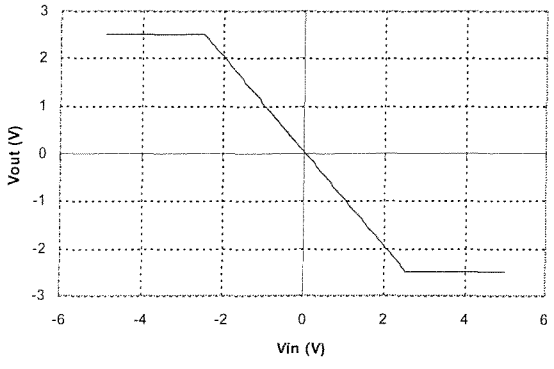


Figure 12: Inverting Amplifier Input Output Voltage for a fault free behavioural model

In the fault I case, the output voltage is a stuck at voltage, with the values -2.14 at -2.5V input to -2.11 at +2V. The region 2V to 2.5V is slightly steeper, with the variation -2.11 to -2.06 for the input range 2V to 2.5V. Therefore to get a highly accurate mapping of output voltage behaviourally would require a PWL model. However, in this case the fact that the model is exhibiting a stuck at fault is adequate, and is accurate to within 5%. To get equation (1) to exhibit this behaviour, m=-1 (cancelling out the input voltage terms) and, k is just the value of stuck at voltage, which is probably best matched at vin=0, $vout$=-2.13V, therefore k=2.13. The simulation result of the behavioural model for this fault type is given in Figure 13.

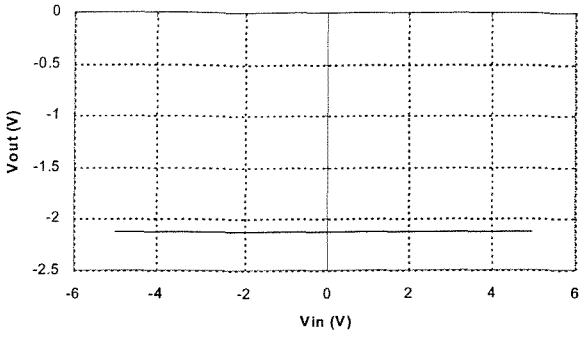


Figure 13: Inverting Amplifier Input Output Voltage for a fault I behavioural model

In the type II faults, the opamp works almost correctly, but the input offset is much higher than normal (10mV), causing a slight offset in the output. This manifests itself with a slight offset in the output voltage, leading to early saturation on one side of the output voltage swing.

In the type III fault case, when the input voltage is greater than zero, the inverting opamp circuit works correctly, but when vin<0, the circuit behaviour turns into non-inverting. In terms of the equation (1), this implies that k=0 and that m is 0 for vin>0 and for vin<0, m=-2. This is summarised in equation (2).

$$vout = \begin{cases} A_{cl} vin & vin > 0 \\ A_{cl}(vin + m * vin) & vin < 0, m = -2 \end{cases} \qquad (2)$$

This discontinuous behaviour cannot be modelled using the simple behavioural model previously given, and a modification was made in the model to include this change in behaviour as shown in Figure 14.

```
if (vin <0) then
    fos := -2.0*vin;
else
    fos := 0.0;
end if;
```

Figure 14: Modification to the behavioural model for type III faults

Using this modified model, the resulting behaviour can be seen in figure 15.

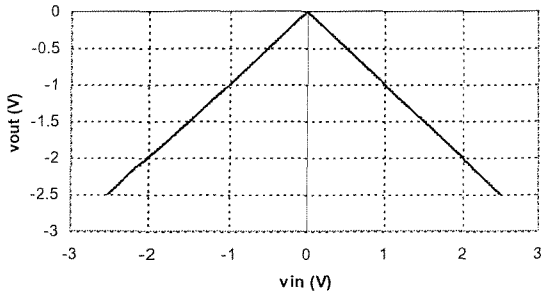


Figure 15: Inverting Amplifier Input Output Voltage for a fault III behavioural model

The same behavioural model can also be used for the non-inverting closed-loop case. In this configuration, the transistor level behaviour can be replicated using the parameters m=0 and k=0 with A_{CL}=2 ($A_{CL} = 1 + R_2 / R_1$ for the non-inverting opamp). On inspection of the transistor level results for the fault IV case, it is clear that a PWL approximation is required to provide a realistic match with a behavioural model. The closed loop model was therefore modified using a PWL voltage offset. The resulting change in behaviour is subtle, but is a slight non-linearity introduced on the output voltage.

4 Open Loop Behavioural Modelling

One drawback with the closed loop model is the restriction on the topologies that can be simulated. If extra components are added into the feedback loop for example, then a complete re-characterisation is needed. To combat this, therefore, a modified fault offset voltage model was created that could be connected to the input of the opamp creating the required fault offset voltage, while allowing an arbirtrary connection to the opamp externally.

This model can then be connected in series with any behavioural opamp model with the option of either specifying specific fault types, or defining the m & k parameters directly. The advantage of using this type of approach becomes clear when more complex circuits are tested such as the IEEE Mixed-Signal Benchmark Biquad filter. Obviously, the opamps are not the simple buffers previously analysed using the closed loop model, and as such the open loop fault model becomes an ideal approach to simulating faults in this type of circuit.

5 Summary of the results

It is clear from the simulation results that there is a good correlation between the transistor level and behavioural level models. The real benefit for multiple simulations depends also on the simulation times in each case, and these are summarised in table 1. For each case a number of runs (3) was taken in each case and the average simulation time (CPU seconds) was recorded (Comparison using Saber).

Circuit configuration	Transistor level Simulation Time (s)	Behavioural Simulation Time (s)
Inverting Fault Free	1.88	0.1
Inverting Fault I	1.78	0.1
Inverting Fault II	1.81	0.1
Inverting Fault III	1.89	0.15
Non-Inverting Fault Free	1.89	0.1
Non-Inverting Fault IV	1.89	0.45
Biquad Filter (a)	11.1	2.35
Biquad Filter (b)	93.2	2.49

Table 1 : Comparison of Simulation Times

In most of the opamp test cases, the speed up in using behavioural models was 18 times. This was slightly reduced in the Fault III and Fault IV models due to the extra solution time required to deal with the more complex PWL characteristic in the model. For the Biquad filter in case (a) the input voltage was half the supply voltage, and in case (b) the input was full scale. It is clear from the difference in simulation times that the relative merits of the MOS and Behavioural models may depend significantly on the operating region of the devices. If the devices are pushed into their non-linear regions, then the solution time may drastically increase as is the case here (x9). It is interesting to note that the behavioural model simulation time is almost exactly the same in both cases implying that the model is robust and can handle the limiting to the supply aspects without undue convergence difficulties.

Complete model listings will be provided for all the models described, and results presented for all the fault conditions in the final paper. Space restrictions preclude that in this shorter version.

6 Conclusions

In this paper, a method of implementing fault behavioural models for operational amplifiers has been presented. Previous work has been extended to cover both the open and closed loop configurations allowing greater flexibility in the application of the fault models in the general case. Results show a good correlation between transistor and behavioural models at all stages, with a corresponding improvement in simulation times.

7 References

[1] Voorakaranam R. and Chatterjee A., "Hierarchical specification driven analogue fault modeling for efficient fault simulation and diagnosis", *IEEE International Test Conference*, 1997, pp903-912

[2] Perkins A.J., Zwolinski M.J., Chalk C.D. & Wilkins B.R., "Fault Modeling and Simulation using VHDL-AMS", *Analogue Integrated Circuits and Signal Processing*, Vol. 16, 1998, pp141-155

[3] Pan C.Y. & Cheng K.T., "Fault Macromodeling for Analogue/Mixed Signal Circuits", *IEEE International Test Conference*, 1997

[4] Milor L. & Visvanathan V., "Detection of catastrophic faults in analogue integrated circuits", *IEEE Transactions on Computer Aided Design (CAD)*, Vol. 8, No. 2, 1989, pp114-130

[5] Sounder T.M. & Stenbakken G.N., "A comprehensive approach for modeling and testing analogue and mixed signal devices", *Proceedings of International Test Conference*, 1990, pp169-176

[6] Nagi N., Chatterjee A., Balivida A., & Abraham J.A., "Fault based automatic test generator for linear analogue circuits", *Proceedings of International Conference on CAD*, 1993, pp88-91

[7] Kilic Y. & Zwolinski M., "Behavioural/Macro modelling to speed up analogueue fault simulation", ELECO 2001 (to be presented)

[8] MAST Reference Manual, Avant! Corporation , USA

[9] IEEE Standard 1076.1 Analogue and Mixed Signal Extensions to VHDL

[10] Boyle G.A., Pederson D.O., Cohn B.M. & Solomon J.E., "Macromodeling of Integrated Circuit Operational Amplifiers", *IEEE Journal of Solid State Circuits*, 1974, pp353-363

[11] Chang Y., Lee C.L., Chen J.E., & SU C., "A Behaviour Level Fault Model for the Closed-Loop Operational Amplifier", *Journal of Information Science and Engineering*, 2000, Vol. 16, pp751-766

[12] Wilson P.R., Kilic Y., Ross J.N., Zwolinski M., Brown A.D., "Behavioural Modeling of Operational amplfier Faults using analogue Hardware Description Languages (HDLs)", BMAS 2001 (to be presented October 2001).

[13] VeriasHDL, Avanti! Corporation, USA. www.avanticorp.com

[14] HAMSter, SIMEC Gmbh., Germany. www.simec.com

9 REFERENCES

[1] M. Abramovici, M.A. Breuer, A.D Friedman, "Digital Systems Testing and Testable Design", IEEE Press, 1990.

[2] S. D. McEuen, " Reliability benefits of I_{DDQ}", Journal of Electronic Testing: Theory and Applications (JETTA), v 3, n 4, December 1992, pp. 327-335.

[3] http://www.logicvision.com

[4] N. Engin, "Linking Mixed-Signal Design and Test, Generation and Evaluation of Specification-Based Tests", PhD Thesis, University of Twente, Enschede, 2000.

[5] L. S. Milor, "A Tutorial Introduction to Research on Analog Circuit Testing", IEEE Transactions on Circuits and Systems-II, Analog and Digital Signal Processing, Vol. 45, No. 10, October 1998, pp.1398-1407.

[6] M. Zwolinski, et al, "Analogue Circuit Test using RMS Supply Current Monitoring", IMSTW, May 15-18, 1996, pp. 149-154.

[7] J.-J. Tang, Kuen-Jong Lee, and Bin-Da Liu, "A Practical Current Sensing Technique for I_{DDQ} Testing", IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Vol.3, No.2, June 1995, pp. 302-310.

[8] Y. Kılıç and M. Zwoliński, " Testing Analog Circuits by Supply Voltage Variation and Supply Current Monitoring", The 1999 IEEE Custom Integrated Circuits Conference, May 16-19, 1999, pp.155-158.

[9] J. Segura, M. Roca, D. Mateo and A. Rubio, "Built-in dynamic current sensor circuit for digital VLSI CMOS testing", Electronics Letters, 29th September 1994, Vol.30, No.20, pp. 1668-1669.

[10] K.-J.Lee, Kou-Shoung Huang and Min-Cheng Huang, "Low voltage built-in current sensor", Electronics Letters, 10th October 1996, Vol.32, No.21, pp. 1942-1943.

[11] K.-J. Lee and Jing-Jou Tang, "A Built-In Current Sensor Based on Current-Mode Design", IEEE Transactions on Circuits and Systems-II: Analog and Digital Signal Processing, Vol.45, No.1, January 1998, pp. 133-137.

[12] J. B. Kim, S. J. Hong, and J. Kim, "Design of a Built-In Current Sensor for I_{DDQ} Testing", IEEE Journal of Solid-State Circuits, Vol.33, No.8, August 1998, pp.1266-1272.

[13] M. Seevinck, "Analog Interface Circuits for VLSI", in Analogue IC Design: the current mode approach, 1990, Toumazou C., Lidgey F. J., and Haigh D. G. (eds.) Peter Peregrinus Ltd. IEE, London, pp. 451-489.

[14] K. Arabi and B. Kaminska, "Design and Realization of an Accurate Built-In Current Sensor for On-Line Power Dissipation Measurement and I_{DDQ} Testing", International Test Conference, 1997, pp. 578-586.

[15] http://asic.vertical-global.com/

[16] META Software, "HSPICE User's Manual Software For IC Design", META-Software INC., Version 96.1 for HSPICE Release 96.1, 1996.

[17] http://www-device.EECS.Berkeley.EDU

[18] E. Bruls, "Variable Supply Voltage Testing for Analogue CMOS and Bipolar Circuits", International Test Conference, 1994, pp.562-571.

[19] A.K.B A'ain, A.H. Bratt and A.P. Dorey, "Testing analgoue circuits by power supply voltage control", Electronics Letters, 3rd February 1994, Vol.30. No.3, pp. 214-215.

[20] A.K.B. A'ain, A.H. Bratt and A.P. Dorey, "Exposing Floating Gate Defects in Analogue CMOS Circuits by Power Supply Voltage Control Testing Technique", 8th International Conference on VLSI Design, January 1995, pp. 239-242.

[21] A.K.B. A'ain, A.H. Bratt and A.P. Dorey, "Testing Analogue Circuits by AC Power Supply Voltage", 9th International Conference on VLSI Design, January 1996, pp.239-241.

[22] K. G. Nichols, T. J. Kazimerski, M. Zwolinksi, and A. D. Brown, "Overview of SPICE-like circuit simulation algorithms", IEE Proc-. Circ. Dev. Syst. 141, pp. 242-250, 1994.

[23] M. Zwolinski, A. D. Brown, C. D. Chalk, "Concurrent Analogue Fault Simulation", 3rd IMSTW, June 3-6, 1997, Seattle, Washington USA, pp.42-47.

[24] J. Hou and A. Chatterjee, "CONCERT: a concurrent fault simulator for analog circuits", 4th IEEE International Mixed-Signal Testing Workshop, 1998, pp.3-8.

[25] A. Balivada, et al, "A Unified Approach for Fault Simulation of Linear Mixed-Signal Circuits", Journal of Electronic Testing, Theory and Applications 9, 1996 Kluwer Academic Publishers, manufactured in The Netherlands, pp. 29-41.

[26] A. Balivada, J. Abraham, "Fault Modelling and Fault Simulation of Analogue Circuits: A Tutorial", IEEE Mixed Signal Testing Workshop, May 15-18, 1996.

[27] H. Hao and E.D. McCluskey, "On the modelling and testing of gate oxide shorts in logic gates", Procs. on VLSI Systems, November 1991, pp. 161-174.

[28] B. Vinnakota, "Analog Test", Prentice Hall PTR, 1998.

[29] http://www.eas.iis.fhg.de/atg/

[30] D. Majernik, B. Lynch, C. Siegel, D. Teegarden, R. Eram, "Using Simulation to Improve Fault Coverage of Analog Test Program Sets", Proceedings of the 1997 IEEE AUTOTESTCON Conference, 1997, pp. 371-375.

[31] M. Ismail, T. Fiez, Analog VLSI Signal and Information Processing, McGraw-Hill, Inc., 1994.

[32] C.D.Chalk, "Novel IC Test Methodologies: Evaluation of AC RMS Supply Current Monitoring", A thesis submitted for the degree of Doctor of Philosophy, February 1997.

[33] D. Majernik, C. Siegel, S. Somanchi, "Behavioral Fault Simulation of Large Mixed-Signal UUTs Using the Saber Simulator", Proceedings of the 1998 IEEE AUTOTESTCON, 1998, pp. 600-605

[34] G. A. Boyle, D. O. Pederson, B. M. Cohn and J. E. Solomon, "Macromodeling of Integrated Circuit Operational Amplifiers", IEEE J. of Solid State Circuits SC-9, pp. 353-363, 1974.

[35] A. J. Perkins, M. Zwolinski, C. D. Chalk and B. R. Wilkins, "Fault Modeling and Simulation Using VHDL-AMS", Analog Integrated Circuits and Signal Processing 16, Kluwer Academic Puclishers, Boston, Manufactured in The Netherlands, 1998, pp.141-155.

[36] C. Chalk and M. Zwolinski, "Macromodel of CMOS operational amplifier including supply current variation", Electronics Letters 31, 1995, pp. 1398-1400.

[37] P. Mandal, V. Visvanathan, "Macromodeling of the AC Characteristics of CMOS Op-Amps", IEEE 1993 Conference on Computer Aided Design, Digest of Technical Papers, 1993, pp.334-339.

[38] B. Al-Hashimi, "Behavioural Simulation of Filters", IEE Colloquium on Analogue simulation: the dream & the nightmare, at Savoy place on Friday, 3 November 1995, pp. 51-55.

[39] http://www.planetee.com

[40] S.Tabatabaei, A.Ivanov, "Schematic-Based Fault Generation, presented at 4th IMSTW, June 1998.

[41] M. Sachdev, "Realistic defect oriented testability methodology for analog circuits ", Journal of Electronic Testing: Theory and Applications (JETTA), v 6, 3, June 1995, pp. 265-276.

[42] C. F. Hawkins, et al, "Quiescent Power Supply Current Measurement for CMOS IC Defect Detection", IEEE Trans. On Industrial Electronics, 1989, vol.36, No.2, pp.211-218.

[43] J. P.M. van Lammern, "ICCQ: a Test Method for Analogue VLSI Based on Current Monitoring", IMSTW June 8-11, 1998, pp.169-173.

[44] http://developer.intel.com/technology/itj/q21998/articles/art_1.htm

[45] http://www.electronicnews.com/news/4366-235NewsDetail.asp

[46] J. Johnson, "Is DfT right for you?", ITC International Test Conference, 1999, pp. 1090-1097.

[47] J. Turino, "Design for Test and Time to Market – Friends or Foes", ITC International Test Conference, 1999, pp. 1098-1101.

[48] A. L. Crouch, M. Pressly, J. Circello, "Testability Features of the MC68060 Microprocessor", International Test Conference, 1994, pp. 60-69.

[49] M. Soma, "A Design-For-Test Methodology For Active Analogue Filters", Digest of papers - International Test Conference, September 1990, pp. 183-192.

[50] A.H. Bratt, et al, "A Design-For-Test Structure for Optimising Analogue and Mixed Signal IC Test Proceedings of European Design and Test Conference, 1995, pp. 24-33.

[51] T. Olbrich, et al, "Design-for-Test (DfT) Study on a Current Mode DAC", IEE Proceedings: Circuits, Devices and Systems, v 143, n 6, Dececmber 1996, p 374-379.

[52] http://grouper.ieee.org/groups/1149/4/index.html

[53] N. Kelly, "BIST vs. ATE for testing System-on-a-chip", ITC International Test Conference, 1998, pp. 1147.

[54] A.A. Hatzopoulos, et al, "A Complete Scheme of Built-In Self-Test (BIST) Structure for Fault Diagnosis in Analogue Circuits and Systems", IEEE Transactions on Instrumentation and Measurement, Vol.42, No.3, June 1993, pp. 689-694.

[55] L.T. Wurtz, "Built-In Self Test Structure for Mixed-Mode Circuits", IEEE Transactions on Instrumentation and Measurement, Vol.42, No.1, February 1993, pp. 25-29.

[56] M.J. Ohletz, "Hybrid Built-In Self-Test (HBIST) for Mixed Analogue/Digital Integrated Circuits", Proc. 2nd European Test Conference 1991, Munich, April 1991, pp. 307-316.

[57] C.L. Wey, "Built-In Self-Test (BIST) Structure for Analogue Circuit Fault Diagnosis", IEEE Transactions on Instrumentation and Measurement, Vol.39, No.3, June 1990, pp. 328-336.

[58] http://www.computer-design.com/Editorial/1997

[59] K. Arabi, B. Kaminska, "Testing Analog Integrated Circuits Using Oscillation-Test Method", IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 16, No. 7, July 1997, pp. 745-753.

[60] P. N. Variyam, A. Chatterjee, N. Nagi, "Low-Cost and Efficient Digital-Compatible BIST for Analog Circuits Using Pulse Response Sampling", 15th IEEE VLSI Test Symposium, 1997, pp.261-6.

[61] M. F. Toner and G. W. Roberts, "On the Practical Implementation of Mixed Analog-Digital BIST", Proceedings of the IEEE Custom Integrated Circuits Conference, Santa Clara, California, May 1995, pp. 525-528.

[62] J.L. Huang and K.-T. Cheng, "A Sigma-Delta Modulation Based BIST Scheme for Mixed-Signal Circuits", Proceedings of Asia and South Pacific Design Automation Conference, Jan. 2000.

[63] M. Soma, "Automatic test generation algorithms for analogue circuits", IEE Procs. – Ciruits Devices Syst., Vol. 143, No. 6, December 1996, pp.366-373.

[64] A. Richardson, A. Bratt, I. Baturone, and J.L. Huertas: "The Application of I_{DDX} Test Strategies in Analog and Mixed Signal Ics", IEEE Int. Mixed Signal Testing Workshop, 1995, Grenoble, France, pp. 44-49.

[65] K.R. Eckersall, P.L. Wrighton, I.M. Bell, B.R. Bannister, and G.E. Taylor, "Testing Mixed Signal ASICs through the use of Supply Current Testing", Proc. 3rd European Test Conf., Rotterdam, The Netherlands, April 1993, pp.385-391.

[66] M. Renovell, F. Azais, and Y. Bertrand, "On-chip signature analyser for analogue circuit testing", Electronics Letters, 1996, Vol.32, No.24, pp. 2185-2186.

[67] B. Johnson, "Built-in Current Sensor for Analogue & Mixed-Signal Circuits, MSc thesis, University of Southampton, 1996.

[68] P. E. Allen and D. R. Holberg, "CMOS Analog Circuit Design", Holt, Rinehart and Winston, 1987, New York.

[69] A. Yildiz, "Implementation of the Built-in Current Sensor for Analogue Circuits", an MSc dissertation, Dept. of ECS, University of Southampton, October 1999.

[70] B. Razavi, "Design of Analog CMOS Integrated Circuits", McGraw-Hill, 2000.

[71] http://hyperphysics.phy-astr.gsu.edu/hbase/electric/filcap.htm

[72] D. A. Johns, K. Martin, "Analog Integrated Circuit Design", John Wiley & Sons, Inc., 1997.

[73] http://www.ee.washington.edu/mad/benchmarks/ benchmarks.html

[74] http://www.Tektronix.com/Measurement/scopes

[75] D. Ager and J. Henderson, "The use of marginal voltage measurements to detect and locate defects in digital microcircuits", IEEE Reliability Physics Symposium 1981, 1981, pp.139-148.

[76] D. Ager, C. Cornwell and I. Stanley, "The application of marginal voltage measurements to detect and locate defects in digital microcircuits", Microelectronics and Reliability, vol. 22, no. 2, 1982, pp. 241-264.

[77] J. Thompson, T. Rogers and R. Galey, "The use of marginal voltage analysis as a screening tool for increased integrated circuit reliability", Journal of Electrical and Electronics Engineering, Australia, vol. 5, September 1985, pp. 235-240.

[78] H. Hao and E.J. McCluskey, "Very-low-voltage testing for weak CMOS logic Ics", Proceedings of IEEE International Test Conference, 1993, pp. 275-284.

[79] A. J. Perkins, "Structural Testing And DFT Insertion For Analogue And Mixed Signal Integrated Circuits", PhD thesis, University of Southampton, 1999.

[80] I. M. Bell, K. R. Eckersall, S. J. Spinks, G. E. Taylor, "Fault Oriented Test and Fault Simulation of Mixed Signal Integrated Circuits", Proc. of IEEE International Symposium on Circuits and Systems, Seattle, Washington, April 1995, pp.389-392.

[81] Y. K. Malaiya, A. P. Jayasumana, Q. Tong, S. M. Menon, "Enhancement of Resolution In Supply Current Based Testing for large ICs", Bridging Faults and IDDQ Testing, IEEE Computer Society Press, Los Alamitos, California, 1992, pp 40-45.

[82] V. Litovski and M. Zwolinski, "VLSI Circuit Simulation and Optimization", Chapman & Hall, 1997.

[83] J. Vlach, K. Singhal, "ccomputer methods for circuit analysis and design", Van Nostrand Reinhold, New York, 1983.

[84] K. E. Brenan, S. L. Campbell, and L. R. Petzold, "Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations", Philadelphia, PA: SIAM, 1996.

[85] W. H. Press, S. A. Teukolsky, W. T. Vetterling, B. P. Flannery, "Numerical Recipes in C", Second Edition, Cambridge University Press, 1997.

[86] E. Christen, and K. Bakalar, "VHDL-AMS, A Hardware Description Language for Analog Applications", IEEE Transactions on Circuits and Systems-II: Analog and Digital Signal Processing, Vol.46. No.10, October 1999, pp.1263-1272.

[87] M Zwolinski, "Relaxation Methods for Analogue Fault Simulation", Proc. 20th Int. Conf. On Microelectronics, (MIEL'95), Vol.2, Nis, Serbia, 12-14 September 1995, pp.467-471.

[88] http://www.cadence.com/datasheets/spectre_cir_sim.html

[89] L. Nagel, "SPICE2: A Computer Program to Simulate Semiconductor Circuits", Memorandum No. ERL-M520, Electronics Research Laboratory, College of Engineering, University of California, Berkeley, 1975.

[90] Mixed-Signal Simulation Overview, Analogy Inc., 1996-1999.

[91] E. L. Acuna, et al, "Simulation Techniques for Mixed Analog/Digital Circuits", IEEE Journal of Solid-State Circuits, 1990, Vol.25, No. 2, pp. 353-363.

[92] Z. R. Yang and M. Zwolinski, "Fast, Robust DC and Transient Fault Simulation of Nonlinear Analogue Circuits", Design, Automation and Test in Europe Conference (DATE99), Munich, Germany, March 9-12, 1999, pp. 244-248.

[93] C.-H. Chen, *Statistical Pattern Recognition*, Hayden Book Company, Inc., Rochelle Park, New Jersey, 1973.

[94] M. W. Tian, C.-J. R. Shi, "Efficient DC Fault Simulation of Nonlinear Analog Circuits", Design, Automation and Test in Europe Conference (DATE98), Paris, France, Feb. 23-26, 1998, pp. 899-904.

[95] C.-Y. Pan and K.-T. Cheng, "Fault Macromodeling for Analog/Mixed-Signal Circuits", IEEE International Test Conference, ITC'97, 1997, pp. 913-922.

[96] G. Casinovi and A. Sangiovanni-Vincentelli, "A Macromodeling Algorithm for Analog Circuits", IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 10, No. 2, February 1991, pp. 150-160.

[97] M. E. Brinson, D. J. Faulkner, "Modular SPICE macromodel for operational amplifiers", IEE Proc.- Circuits Devices Syst., Vol. 141, No. 5, October 1994, pp. 417-420.

[98] M. E. Brinson, D. J. Faulkner, "A SPICE Noise Macromodel for Operational Amplifiers", IEEE Transactions on Circuits and Systems-I: Fundamental Theory and Applications, Vol. 42, No. 3, 1995, pp. 166-168.

[99] G. Krajewska and F. E. Holmes, "Macromodeling of FET/Bipolar Operational Amplifiers", IEEE Journal of Solid-State Circuits, Vol. SC-14, No. 6, 1979, pp. 1083-1087.

[100] C. Turchetti and G. Masetti, "A Macromodel for Integrated All-MOS Operational Amplifiers", IEEE Journal of Solid-State Circuits, Vol. SC-18, 1983, pp. 389-394.

[101] M. E. Brinson, D. J. Faulkner, "SPICE macromodel for operational amplifier power supply current sensing", Electronics Letters, Vol. 30, No. 23, 10th November 1994, pp.1911-1912.

[102] R. V. Peic, "Simple and Accurate Nonlinear Macromodel for Operational Amplifiers", IEEE Journal of Solid-State Circuits, Vol. 26, No. 6, 1991, pp. 896-899.

[103] B. Perez et al, "A New Nonlinear Time-Domain Op-Amp Macromodel Using Threshold Functions and Digitally Controlled Network Elements", IEEE Journal of Solid-State Circuits, Vol. 23, No. 4, 1988, pp. 959-971.

[104] A. I. Kayssi, K. A. Sakallah, "Macromodel Simplification Using Dimensional Analysis", 1994 Int. Symp. On Circuits and Systems, 1994, pp. 335-338.

[105] G. Casinovi, "Multi-Level Simulation of Large Analog Systems Containing Behavioral Models", IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems, Vol.13, No.11, November 1994, pp. 1391-1399.

[106] C. G. Broyden, et al, "A class of methods for solving nonlinear simultaneous equations", Mathematics of Computation, vol. 19, no. 92, 1965, pp. 577-593.

[107] Y.-J. Chang et al, "A Behavior-Level Fault Model for the Closed-Loop Operational Amplifier", Journal of Information Science and Engineering, Vol. 16, No. 5, September 2000, pp. 751-766.

[108] E. Bruls, et al , "Analogue fault simulation in standard VHDL", IEE Proc. Circuits Devices Syst., Vol. 143, No. 6, December 1996, pp.380-385.

[109] M. Zwolinski, Z. R. Yang, T. J. Kazmierski, "Using robust adaptive mixing for statistical fault macromodelling", IEE Proceedings: Circuits, Devices and Systems, v 147, n 5, October 2000, pp. 265-270.

[110] http://www.ednmag.com

[111] VHDL Language Reference Manual, IEEE Standard 1076-1993.

[112] Standard Description Language Based on the Verilog™ Hardware Description Language, IEEE Standard 1364-1995.

[113] www.mentor.com

[114] www.cadence.com

[115] www.verilog.com

[116] www.analogy.com

[117] www.hamster-ams.com

[118] 1076.1 Design Objective Document V 2.3 (1995) http://vhdl.org/vi/analog

[119] 1076.1 Design Objective Rationale V 1.2 http://vhdl.org/vi/analog

[120] http://www.eda.org/analog/ftp_files/documentation/tutdac99.pdf