University of Southampton

Switched-Current Filters and Phase-Locked Loops: Methods and Tools

by

Reuben Wilcock

A thesis submitted for the degree of

Doctor of Philosophy

in the

Faculty of Engineering and Applied Science

School of Electronics and Computer Science

November 2004

UNIVERSITY OF SOUTHAMPTON

ABSTRACT

School of Electronics and Computer Science

Faculty of Engineering and Applied Science

Switched-Current Filters and Phase-Locked Loops: Methods and Tools

By Reuben Wilcock

The switched-current (SI) technique has started a new era in analogue sampled-data signal processing where the benefit of requiring no linear floating capacitors and the suitability for low-voltage operation facilitates mixed-signal design on a standard digital CMOS process. This thesis considers the analysis, design automation and realisation of two fundamental analogue blocks, filters and phase-locked loops (PLLs), using SI technology.

A systematic design flow, from specification to layout, for SI filters employing the wave synthesis technique is presented. A key feature of the flow is a novel power-aware scaling procedure which simultaneously reduces the filter power consumption and total harmonic distortion. A computer aided design (CAD) methodology called AutoSIF has been developed to automate the filter design flow and facilitate rapid generation of SI wave filter circuits. PLLs are employed in numerous applications ranging from clock recovery to demodulation and frequency synthesis. Despite the possible advantages of applying the SI technique to PLLs, there has been very little research in this area. This thesis describes the methodical design of SI PLLs and proposes a novel 2^{nd} order architecture that does not require a separate phase detector, leading to a more compact implementation than conventional approaches. Theoretical analysis and transistor level design procedures are presented for the proposed PLL architecture and a further CAD methodology, AutoPLL, has been developed to automate and support the associated design flow.

Simulation results based on foundry BSim3v3 models are provided for the designed SI filters and PLLs. To analyse the practical performance of the power-aware filter design flow, a prototype chip has been fabricated and measured results show close agreement with theoretical analysis and simulation. Two PLL case studies are presented including a 10MHz frequency shift keying (FSK) demodulator and 500MHz frequency synthesiser which demonstrate that the proposed SI PLL architecture is capable of producing low power designs of comparable performance to the commercial state of the art.

Acknowledgements

I would like to express my sincere gratitude to my supervisor, Bashir Al-Hashimi, without whom this project would have not been possible. I am also extremely grateful for the EPSRC funding I have received during the past three years and the research facilities provided by the School of Electronics and Computer Science at Southampton University.

I would particularly like to thank Neil Ross for many constructive and thoroughly enjoyable 'Friday afternoon chats' concerning analogue circuit design. Thanks are also due to Peter Wilson for his help with matters of simulation and modelling and Mark Zwolinski for his useful comments at the interim review stages. John Hughes also deserves thanks for his valuable emails concerning memory cell design as does Andrew Becket for the amazing dedication he devotes to supporting Cadence users through the unofficial online forum. The help and encouragement from friends in particular Paul, Richard, Mehdi, Manoj and Mauricio has been most appreciated.

If one person has contributed most to helping me through the last three years it has undoubtedly been my girlfriend, Natasha. Thanks for putting up with my long working hours and as much time as I put into this thesis, I never stopped being distracted by thinking of you.

Finally, I would like to thank my parents for the unconditional support they have always given me for the choices I have made during my life, and I would like to dedicate this thesis to them.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Since the concept for the integrated circuit (IC) was first published by Geoffrey W.A. Dummer in May 1952, there has been a phenomenal increase in the level of circuit design integration. Today it is commonplace for ICs to contain tens of millions of transistors and perform a multitude of highly complex functions. At present, CMOS is the dominant process technology for the implementation of state of the art consumer and industrial products. CMOS process development is primarily driven by the speed, area and power requirements of digital circuitry and in particular microprocessors, and DRAM memories. Unfortunately, these requirements conflict with those for analogue functionality in several key areas:

Integration of passive components: Conventional monolithic analogue designs require high quality integrated resistors and capacitors. Only one polysilicon (poly) layer is necessary for implementing digital gates, yet two poly layers are required to integrate linear floating capacitors. This second poly layer is a commercially expensive process option, increasingly resisted by semiconductor manufactures whose profit margins are already small [1]. Furthermore, modern digital processes are optimised for low capacitance per unit area in order to achieve high speed operation, which is in direct contrast to the high capacitance per unit area desirable for analogue integrated capacitors.

Supply voltage reduction: The shrinking feature sizes of modern processes irrefutably allows a higher level of integration, however, low supply voltages are required to prevent breakdown of the resulting small junctions and thin oxides. This reduced voltage headroom directly impacts on the achievable dynamic range of voltage mode analogue circuits [2].

Threshold voltage scaling: Cascoding is a common analogue technique which greatly improves the output conductance of short channel length devices. However, threshold voltages of modern processes do not scale linearly with supply voltage reductions and this limits the number of transistors which can be stacked between the power and ground rails whilst being maintained in the saturation region [1, 3]. The possibility of employing cascoding techniques is therefore significantly reduced.

In addition to the above problems, it is becoming increasingly clear that the gaining popularity of a System on Chip (SoC) design flow is presenting a further significant challenge for analogue designers. SoC involves the integration of complex systems consisting of many functional blocks, called Intellectual Property (IP) cores, onto a single chip. Many mixed-signal SoCs are already in use in multimedia and consumer products, and industrial devices such as sensor interfaces. However, unlike a standard digital flow, analogue functions are tied so closely to process technology and vendor, that design iterations can be lengthy and process migrations an arduous and costly task. Limited innovation in analogue design methods has been a primary cause for the lack of customizable analogue solutions. Traditionally, analogue circuit design has been performed on a full-custom one-off basis, with little or no design reuse, and no set design methodology or supporting tools. With SoC in mind, methods and tools to facilitate the efficient design of major analogue blocks are therefore essential [2].

The problem of analogue circuit realisation in a SoC environment is becoming so widely recognised that according to the 2003 International Technology Roadmap for Semiconductors one of the biggest challenges in SoC today is integrating high-performance and low-cost analogue/mixed-signal solutions, with particular issues being passive component integration and reduced power supplies [1]. One could question the part analogue circuits have to play in the SoC future given such a

number of challenging issues. However, despite becoming more peripheral in nature, analogue functionality still continues to fulfil important roles:

Real world interfacing: The need for real world interfaces is ever present and growing and so an infinitum of real world analogue signals will always exist to process, condition and generate. This ensures that at the very least, analogue filters, amplifiers, data converters and phase-locked loops will always be required, if only to facilitate the subsequent digital processing of these signals. Furthermore, many communications applications will always rely on analogue functionality due to the nature of the signals involved.

Speed, area and power considerations: Analogue design techniques are becoming increasingly considered in place of their digital counterparts to cope with the demands of higher speed operation, increased portability and lower power. For example, analogue implementations of some processing tasks require less area or power than their digital counterparts, and in these cases the analogue alternatives may be favoured.

The key to the future of analogue circuit design for SoC is to determine a circuit design technique suitable for this environment. The motivation of this thesis is to address the requirements for implementing analogue functionality on SoCs through providing design methodologies and tools for two major analogue building blocks: filters and phase-locked loops.

1.1 The switched-current technique

The switched-current (SI) technique is a strong candidate to satisfy the requirements for designing successful analogue circuits for use in present and future SoCs. Switched-current is a current domain sampled data signal processing technique originally advocated in 1972 for photodiode applications [4]. The principle, shown in Figure 1.1, is as follows: when ϕ is closed the transistor's drain current, I_D, will follow the changes in the gate voltage, V_{gs}, (normal square law models), but when ϕ is open, the MOS transistor will maintain its drain current due to the charge stored on its gate source capacitance, C_{gs} [5].

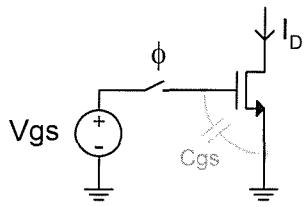


Figure 1.1 The switched-current principle

The application of SI techniques to signal processing was first demonstrated with a sampled data filter by Hughes *et al* in 1989 [6, 7]. Using the principle shown in Figure 1.1, the first generation SI 'memory cell' was created in 1990 [7, 8], and a growing interest in SI followed. The first generation memory cell, shown in Figure 1.2(a) uses two PMOS current sources to bias the NMOS transistors such that an input current of both positive and negative value can flow. When ϕ_1 is high the circuit's function is that of a current mirror and the current I_{in} flows into the cell, summing with the bias current, J. The sum current is mirrored by the output transistor, producing an equal but opposite copy of I_{in} at the output I_{out}:

$$I_{D1} = J + I_{in} \qquad I_{D2} = I_{D1} \tag{1.1}$$

$$I_{out} = J - I_{D2} \qquad I_{out} = -I_{in} \tag{1.2}$$

When the switch is released, the charge stored on the output NMOS transistor's parasitic capacitance holds its gate voltage, hence maintaining the output current for as long as the switch is open. This behaviour is equivalent to a current-mode track and hold function, and can be seen in Figure 1.2(b). Cascading two SI memory cells gives a SI delay cell, which is equivalent to z^{-1} in the sampled (z) domain. The delay cell can be used to implement more complex circuit blocks including, most notably, filters [8-15] and analogue to digital converters [16-21]. A SI Viterbi decoder [22], oscillator [23] and 1st order PLL [24] have also been reported.

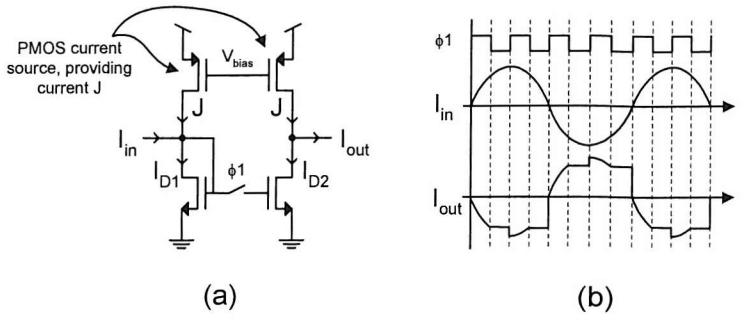


Figure 1.2 First generation SI memory cell (a) and example signals (b)

The first generation memory cell was subsequently improved with the second generation cell, which achieves current memory with a single transistor and is shown in Figure 1.4(a) [5]. Non-ideal effects, such as mismatch, settling, conductance ratio and clock-feedthrough errors, limited the performance of the first and second generation SI cells, which was less than state of the art switched capacitor (SC) equivalents. Clock feedthrough was the main performance limiter and a brief explanation is shown in Figure 1.3 which illustrates a cross-section of the memory cell sampling switch.

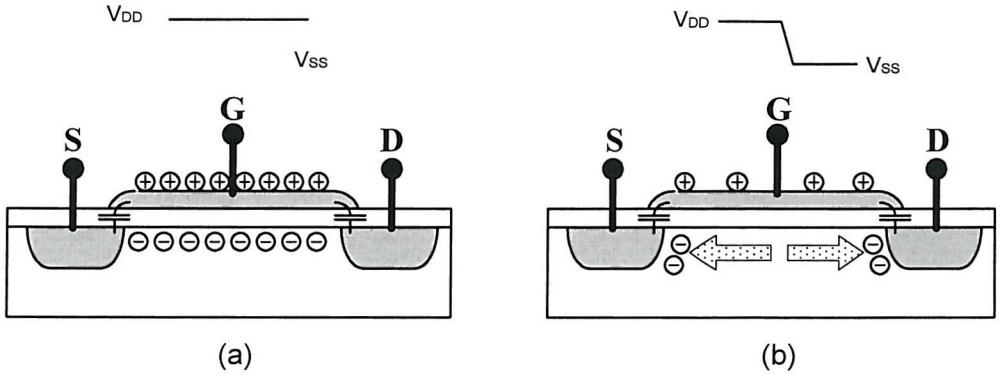


Figure 1.3 The cause of clock feedthrough errors in SI circuits

When the sampling switch is turned off at the end of the sampling phase, the channel charge in the switch transistor is dissipated through its drain, source and substrate connections and hence some of this charge is added onto the memory transistor's gate capacitance. In addition, the rapidly changing switch gate voltage forces current through the switch's gate source overlap capacitance into the memory transistor's gate capacitance. These charge transfers affect the voltage level on the memory

transistor gate and hence an error in the memorised current results. The magnitude of this error will depend largely on the ratio between the switch gate area and the memory transistor gate area, and for this reason switches are made as small as possible. A number of circuit techniques have also been proposed to alleviate the problem of charge injection:

Dummy switches: A transistor half the size of the switch transistor with drain and source shorted is placed between the switch and the memory transistor gate. This dummy switch is controlled by an inverted clock such that when the normal switch turns off the dummy switch turns on and collects the injected charge. The effectiveness of this technique depends on the accuracy of the inverted clock phase and the sizing of the dummy switch.

Transmission gates: Transmission gates containing an NMOS and PMOS switch in parallel can also be used to reduce charge injection. Through careful switch sizing, all of the injected charge from one phase can be absorbed by the complementary switch. Similar to the dummy switch technique, the clock phases and switch sizing is critical to obtain good results using this approach.

Modern iterations of the first and second generation memory cell are showing good promise in terms of high accuracy and low power [25]. The most notable of these include the S^2I memory cell [26] and class AB memory cell [27]. Figure 1.4 shows the second generation, S^2I and class AB memory cell, and the latter two will be discussed in detail where they are used later in the thesis.

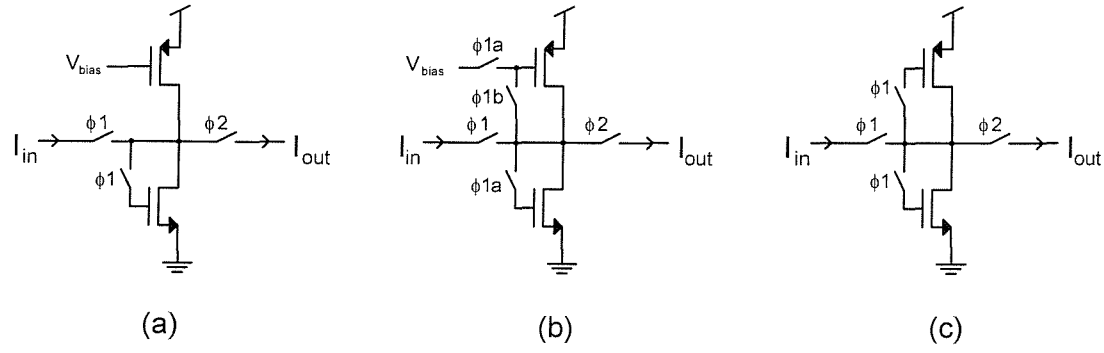


Figure 1.4 Improved memory cells: second gen. (a) S^2I (b) and Class AB (c)

A direct comparison of SI with SC in [28], with past and future processes, showed how the general 'figure of merit' of SI is likely to stay constant over future process advances with that for SC dropping below this within five years. This trend exists because SI circuits offer a number of unique advantages which place themselves as excellent candidates for present and future mixed signal SoCs:

- SI circuits do not require linear passive components and can therefore be designed for any CMOS process having only the most basic digital layers.
- SI designs can function at low voltages, ensuring suitability for future processes [28].
- Consisting mainly of simple current mirrors, SI circuits have regular structures making them ideal for automated generation.
- The frequency of operation of SI circuits is directly related to transistor time constants and hence reducing feature sizes result in increasing potential for high frequency operation [29].

1.2 Thesis contributions and organisation

The body of this thesis comprises three chapters, two, three and four which represent the main contributions of this research. Each chapter is largely self contained and as such incorporates relevant background material and a literature review. The driving motivations and subject matter for these chapters are summarised as follows.

Chapter 2: Filters are a key analogue block and have been the focus of much of the work in SI. Two SI filter design approaches have mainly been used in the literature: integrator based designs, which follow a similar synthesis procedure as SC filters and wave analogue filter designs, which are based on the wave digital filter (WDF) method developed by Fettweis [30]. Both of these approaches have shown good success in the literature, with the majority of the research in the last 10 years being focussed on the integrator based approach [13-15, 31, 32]. SI wave filters received a great deal of interest when they were pioneered in the early 1990s when it was indicated that they were a viable alternative to their integrator based counterparts [10-12, 33, 34]. Chapter 2 presents a systematic design flow for SI filters employing the wave synthesis technique, detailing all aspects from specification to layout. Since power consumption is of such importance in modern system designs, there is a clear

argument for determining whether a procedure to save power could be developed during the wave filter design process. A key feature of the described design flow is a novel power-aware scaling procedure which simultaneously reduces power consumption whilst improving total harmonic distortion (THD). To analyse the practical performance of the filter design flow and power-aware design method, a prototype chip has been fabricated. The measured results taken from this device show excellent agreement with theoretical analysis and simulation. The fabricated wave filter chip operates at a higher cut-off frequency than existing reported fabricated wave filter chips [11], and is based on an elliptic filter response as opposed to the all pole filters in [11].

Chapter 3: A lack of tools and understanding is hindering the use of the SI approach, eminently suitable for system chips. In Chapter 3, it is suggested that a certain degree of design automation, including both tools and cell libraries may be the key to unlocking the potential of SI for SoC applications. By examining the strengths and weaknesses of previous computer aided design (CAD) tools for SI filters and analogue IP cores, a new CAD methodology is proposed which allows rapid generation of the SI wave filters developed in Chapter 2. The proposed methodology, called AutoSIF, includes a carefully developed and parameterised cell library and a tool which automates well captured design steps. The methodology is not only capable of rapidly developing analogue filter cores but also provides good insight into the resulting SI design process. The methodology was used extensively to help with the design of the prototype chip and other filter examples detailed in Chapter 2.

Chapter 4: Phase-locked loops (PLLs) are fundamental building blocks, employed in a vast range of applications spanning from clock recovery, to frequency shift keying (FSK) demodulation and frequency synthesis. To the best of the author's knowledge, there exists only one example of a SI PLL in the literature, which was published in 1997 [24]. PLLs utilising SC filters have been reported in the literature, however, a fundamental limitation is that they require integrated linear capacitors unlike their SI counterparts [35]. Chapter 4 explains how using the SI technique for fully integrated PLLs could offer many advantages, such as low power consumption, high frequency operation and a small silicon area. These advantages along with the lack of previous

work in the area are key motivating factors for developing high-performance SI PLLs. The previous SI PLL work in [24] had limited use since the presented design operated at a relatively low frequency of 1MHz and utilised only a 1st order loop. Most current and future practical PLL applications require 2nd order architectures with much higher frequencies of operation. Chapter 4 describes the methodical design of SI PLLs and proposes a novel 2nd order architecture that does not require the employment of a separate phase detector, as is the case with conventional PLLs. Theoretical analysis and transistor level design procedures are presented, and the effectiveness of the approach is demonstrated with example designs including a 10MHz FSK demodulator and 500MHz frequency synthesiser. The proposed approach is capable of producing designs for practical applications with low power consumption, and of comparable performance to the commercial state of the art.

The novel contributions in Chapters 2, 3 and 4 are summarised in Chapter 5 which also considers further research directions based on the findings in this thesis. During the course of this research, a number of papers have been published, or submitted for publication, in [36-41] and these are listed below.

R. Wilcock and B. M. Al-Hashimi, "**Analogue Filter IP Cores for Design Reuse**", Proceedings of Analog Signal Processing Conference, vol. 1, pp. 2.1-2.6, November 2002.

R. Wilcock and B. M. Al-Hashimi, "**A CAD Methodology for Switched Current Analog IP Cores**", Proceedings of IEEE International Conference on Emerging Technologies and Factory Automation Conference, vol. 1, pp. 434-437, September 2003.

R. Wilcock and B. M. Al-Hashimi, "**Power-Aware Design Method for Class A Switched-Current Wave Filters**", IEE Proceedings - Circuit Devices and Systems, vol. 151, pp. 1-9, February 2004.

R. Wilcock and B. M. Al-Hashimi, "**Power-Conscious Design Methodology for Class-A Switched-Current Wave Filters**", Proceedings of IEEE International Symposium on Circuits and Systems, vol. 1 pp. 225-228, May 23-26 2004.

R. Wilcock and B. M. Al-Hashimi, "**Switched-Current Wave Filter Design: from Specification to Layout**", Submitted to the IEEE Transactions on Circuits and Systems, Part I, June 2004.

R. Wilcock, P. Wilson and B. M. Al-Hashimi, "**A Novel Switched-Current Phase-Locked Loop**", Submitted to the IEEE International Symposium on Circuits and Systems, October 2004.

R. Wilcock, B. M. Al-Hashimi and P. Wilson "**Integrated High Bandwidth Wave Elliptic Low-Pass Switched-Current Filter in Digital CMOS Technology**", Submitted to the IEE Electronics Letters, October 2004.

Chapter 2

Power-Aware Switched-Current Filters

Filters are a key mixed-signal building block and have been the main focus of the switched-current (SI) technique since it was first proposed for signal processing applications in 1989 [6, 7]. Two main SI filter design approaches have been proposed in the literature: integrator based designs, following a similar synthesis procedure to switched capacitor filters and wave analogue filter designs, which were based on the wave digital filter (WDF) method developed by Fettweis [30]. This chapter presents a complete design process from specification to layout for SI wave filters, illustrated through a case study example. In order to facilitate robust, high performance designs, both the ideal high level coefficients and low-level transistor dimensions must be optimised. A key contribution demonstrates that power consumption can be reduced using a novel two stage bias and signal current scaling method, which also improves filter total harmonic distortion. Practical layout techniques are described which address the matching issues prevalent in this class of circuit. The proposed design flow is validated through the design and fabrication of a 3rd order lowpass elliptic case study filter, using a 3.3V 0.6μm CMOS process. The presented filter not only has the highest bandwidth reported to date for fabricated SI wave filters [11], but also exhibits an elliptic response, in contrast to the all-pole wave filters reported in [11].

Section 2.1 provides a review of the previous work in SI filter design and outlines the chosen filter design method. Section 2.2 details wave filter design at the ideal block level including design of the passive prototype ladder filter, wave structure and adaptor coefficient calculation and optimisation. Section 2.3 presents a novel power aware scaling method which not only reduces the distortion caused by large internal signal swings but also reduces power consumption. Transistor level design of the wave filter building blocks is detailed in Section 2.4 where a practical layout strategy is also described. Simulated and measured results for the case study chip are given in Section 2.5 along with further results verifying the power aware scaling method.

2.1 Preliminary review

Section 2.1.1 provides a detailed review of the research activity in SI filter design over the last 17 years. For clarity, a timescale of the most significant developments in this field is shown in Figure 2.1. Section 2.1.2 presents the main advantages of using the wave synthesis technique for SI filter design. Finally, Section 2.1.3 presents the specifications for the case study wave filter developed throughout this chapter.

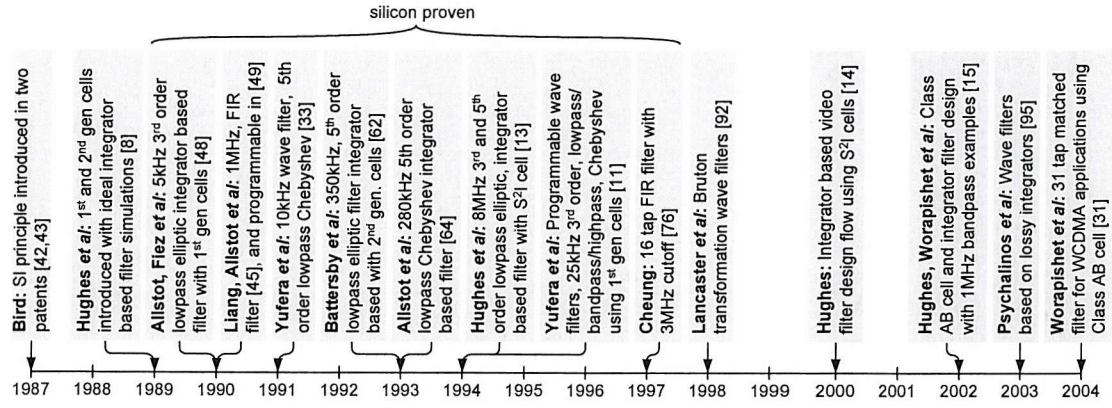


Figure 2.1 Timescale showing major SI filter developments

2.1.1 SI filter review

The SI technique for storing current samples was published by Bird in two patents in 1987 [42, 43]. Hughes, working with Bird at Philips Semiconductors (UK) at the time, adopted the technique but for signal processing applications and published the use of the SI technique for filtering in 1989 [6, 7]. In these publications, the first and

second generation memory cells were introduced and the first generation cell used to implement Euler and bilinear integrators. A test chip was fabricated to verify the operation of these basic cells. Using an in-house CAD tool, Hughes *et al* were able to prove that, at the ideal level, the SI technique could be used to implement a 6^{th} order Chebyshev lowpass filter by cascading second order sections. In the following year (1990), the ideal simulations were extended to model parasitics [8] and a bilinear integrator based on the second generation memory cell was proposed [44]. Encouraged by the potential of SI, these early papers [6-8, 44] fostered an explosion of interest in the SI technique in the 1990s with over 70 papers published concerning SI filter design alone.

A second research group, led by Allstot and Fiez from Oregon State University were also involved in early developments into SI techniques with six publications in the year of 1990 [45-50]. Measured results from the first two fabricated SI filters were presented in [47, 48, 50] by Fiez *et al* which included a 3^{rd} order elliptic and 5^{th} order Chebyshev lowpass response, both with 5kHz cut-off. The filter design technique used was identical to that for switched capacitor (SC) filters, and involved cascading differential integrators. Fiez *et al* also demonstrated lowpass, highpass and bandpass biquad based designs in [46] with simulated results showing cut-offs in the low kHz region. Liang and Allstot also proposed the first SI finite impulse response (FIR) filter in [45] with measured results showing a lowpass cut-off frequency of up to 1MHz. Begisi, Fiez and Allstot extended the FIR filter to allow programmability in [49]. FIR and infinite impulse response (IIR) SI filter realisation continued to be explored by Liu [51] and Song [52] in 1991 and 1992 respectively with low frequency (1-5kHz) filters being demonstrated at transistor level.

Until this point all reported SI filters were either based on integrator or FIR/IIR designs. In 1991/92 the first application of SI to wave filter design was published by Rueda and Yufera *et al* [10, 33]. Wave filters had been reported in detail by Fettweis in 1986 [30] for digital realisation but these did not extend well to voltage mode analogue implementations. However, Yufera and Rueda demonstrated that wave analogue filters were particularly well suited to an implementation in the current mode SI technology. In 1991, they showed how an equivalent wave filter could be

generated from either a passive or microwave starting point with simulated results from a 5th order lowpass Chebyshev filter with 1kHz cut-off. Measured results from a test chip fabricated on a 1.5μm process were published by Yufera *et al* in 1992 and showed an accurate lowpass response at a 10kHz cut-off [33]. This pioneering work on SI wave analogue filters offered an exciting alternative to integrator based filter design and has attracted interest ever since. In 1993, Jonsson *et al* presented the now standard n-port adaptor design in [53] and a clock feedthrough reduction method in [54]. Jonsson later detailed a fabricated lowpass wave analogue filter design based on his n-port adaptors and improved memory cells (no measured results provided) [12].

A number of directions in SI filter design were now emerging including integrator based, FIR/IIR based and now wave based designs. Integrator based filter methods continued to receive the most interest mainly because of their similarity to the mature SC design method, a relationship described by Roberts *et al* in 1991 who proposed a systematic translation between SC and SI filter designs [9]. Later, in 1993, Song and Roberts presented a 5th order elliptic low pass (1kHz cut-off) filter with a new SI integrator configuration [55]. De Queiroz *et al* presented nodal analysis of SI integrator based filters in 1991/93 providing sensitivity analysis of SI filters based directly on passive prototypes [56-58]. This work was supported by a simulator, AZIS which was targeted towards high level analysis. The same author presented a switching sequence shown to reduce switching noise in SI filters in 1993 [59].

Battersby and Toumazou published three papers in 1993-1994 detailing a 5th order lowpass elliptic filter fabricated on a 1.2 μm process [60-62]. The filter was designed with bilinear integrators comprising second generation memory cells and had a 350kHz cut-off frequency. Experimental implementation of a SI filter on a GaAs process was also considered in [61], which appeared to allow much higher frequency designs (1GHz sampling) with a 100MHz lowpass filter cut-off. Allstot and Zele detailed work on a fabricated fully differential 5th order Chebyshev integrator based SI filter in 1993-1994 [63, 64]. The filter was implemented on a 1.2μm process with a low supply voltage of 3.3V and demonstrated a cut-off frequency of 280kHz. By the end of 1993 in light of the considerable interest the SI approach had received, a book summarising the state of the art in SI design was edited by Toumazou [65].

In 1994-1996 de Queiroz *et al* published work on SI bandpass filters [66] bandpass filters from component simulation (Schechtman *et al*) [67] and bilinear ladder filters using Euler integrators [68]. All these filters were supported by simulated results using the ASIZ simulator. An experimental bipolar SI filter implementation was also presented by de Queiroz *et al* in 1996 who concluded that such an implementation was not very useful for practical applications [69]. An interesting novel SI filter design considered in 1995 by Tse *et al* was a median filter [70]. A median filter implements an algorithm to determine the most common sample value from a given set and has applications in image processing for reducing speckle noise. Another interesting extension on SI filter design in 1995-1996 were the programmable integrators introduced by Goncalves *et al* [71, 72]. Despite the useful feature of programmability, the integrators consisted of operational amplifiers and grounded capacitors and so lose some of the advantages of more conventional SI design.

The pioneer of SI signal processing, Hughes, further improved the SI filter state of the art in 1994-1996 with a number of fabricated high performance filter chips [13, 32, 73]. During this period Philips Semiconductor had been working on a CAD tool for the generation of high performance integrator based SI filters as part of the SCADS project. Hughes had developed a double sampling bilinear integrator based on the S^2I memory cell which he presented earlier in 1993 [26]. Fabricated filters presented in [13] included balanced 3^{rd} and 5^{th} order lowpass elliptic filters integrated on a 0.8μm process and both operating with cut-offs in the low MHz region.

The next major advancement on the SI wave analogue filter front was again by Yufera *et al*, in 1994 [11]. A universal filtering function had been developed which could implement different filter functions with a fixed circuit topology. First generation memory cells and high compliance current mirrors were used on a test chip which was fabricated on a 1.6μm process. A number of measured filter responses were given for lowpass, bandpass and highpass filtering functions, with band edges of up to 25kHz. Also published by Yufera *et al* in 1994 was a paper considering mismatch in wave analogue filters, [34], work which was continued in a publication a number of years later in 1997 [29] where it was concluded that clock feedthrough sensitivity was worse for integrator than wave filter based designs.

Further work on FIR/IIR based filters was proposed from 1995-2000. For example, Wang *et al* in 1995 with a simulated second order lowpass IIR filter in [74] and a fully programmable IIR filter topology by Ahmad *et al* in 1997 [75]. A fabricated 16 tap FIR filter on a 0.8μm process was presented by Cheung in [76] for channel equalisation applications and showing a 3MHz cut-off. More recently, a 4 tap 2V programmable FIR filter employing a solution to the problem of switching with low voltages was proposed by Farag *et al* in 1999-2000 [77, 78].

Vega and de Queiroz presented work on adaptive SI filter cells in 1998 and later in 2001 [79, 80]. Sensitivity and error reduction by component swapping was also considered by de Queiroz *et al* in 1999 [81] leading to the conclusion that component swapping can lead to improved sensitivity. In 2001 de Queiroz published work on SI lattice filter design [82] suggesting that lower sensitivity is found by implementing SI filters from unbalanced passive lattice filters. In 2000 Handkiewicz *et al* discussed the application of SI filters to image processing systems (although no systems were implemented) [83], and later in 2002 the same author presented a tool for SI filter design [84]. A number of SI filter CAD tools for integrator based filters were proposed in 2001, including Barau *et al* [85] and Qingyun *et al* [86].

Hughes summarised the contributions of the SCADS project in 2000 detailing a top down design flow for integrator based SI video filters [14]. With Worapishet, he then began work on a next generation of SI filters, using Class AB memory cells. A number of advancements had been made in Class AB memory cell design by Hughes *et al*, which were detailed in a chapter of [87] in 2002 and four papers [15, 88-90] demonstrating the suitability of Class AB memory cells for use in high frequency complex channel filters. Simulated results for 1MHz bandpass filters were given to demonstrate the feasibility of the designs. Later in 2004 Worapishet *et al* published work on a 31 tap matched filter for WCDMA applications again based on Class AB memory cells and using a 0.35μm process [31].

Wave filters based on Bruton transformations were discussed in 1998 by Lancaster *et al* [91] and later in 1999 where they were shown to be especially suitable for higher order filter functions [92]. In 2003, an analysis of mismatch and clock

feedthrough in Bruton filters was given by Xie *et al* [93] who later discussed adaptive wave filters [94]. Wave filters based on microwave prototypes and lossy integrators were presented by Kontogiannopoulos and Psychalinos in 2003 along with a new parallel adaptor which reportedly leads to a simpler overall architecture [95, 96].

2.1.2 Chosen filter methodology

The last section showed that three main methods have been applied to SI filter design: integrator [13, 14], FIR [76, 77], and wave based approaches [11, 12]. Of the three, both integrator and wave designs have the considerable advantage that they are derived from passive reference filters and so inherit an excellent passband sensitivity to component variations. FIR filters [45, 76] are easy to implement in SI with a delay cell and current mirror for every tap, however, not emulating a passive filter leads to inferior designs as compared to integrator [13] or wave [11] based approaches. Furthermore, FIR filters can require many more stages than integrator or wave based filters for an equivalent filter specification.

Although integrator based filters have received most attention in the literature, it is possible that wave filter design may offer the best solution in terms of SI analogue filter cores. Wave digital filters emulate the behaviour of passive lossless filters by transforming passive L and C components into one-port digital elements defined by an incident signal, a reflected signal and a port resistance. Adaptors are used to connect these one-port elements. Wave filters were originally found impractical for implementation in analogue continuous-time or SC technology as the resulting filter structures were too complex. However, the operations required to realise wave filters are summation and multiplication by a constant, these being easily achieved in current-mode. The key advantages of the SI wave filter approach are given below:

- Wave filters are inherently based on the bilinear transform, allowing high frequency filters [33].
- Wave structures are formed by interconnection of easily designed blocks, which are highly suitable for automated generation [12].

- The sum of coefficient values in any adaptor must equal two, meaning that end transistor dimensions do not have to have a one-to-one correspondence with transfer function coefficients unlike integrator based designs, allowing a degree of freedom in the wave coefficient choice [65].
- The freedom in choice of some adaptor coefficients can be exploited to optimise the transistor level design for low sensitivity to process variations and reduced technology grid rounding errors [11].
- Wave filters inherit the low passband sensitivity to component variations enjoyed by the passive filters from which they are derived [33].
- Some variations on the wave filter methodology have been shown to be efficient for designing high order filters [92].

These advantages indicate that further research into the SI wave filter design technique would be of value. While systematic design flows for integrator based SI filters have been addressed in the literature [14] and indeed automated [13, 84-86, 97], wave SI filter design has not received a similar treatment. To facilitate the acceptance and maturity of SI wave filters, detailed flows are needed. The aim of this chapter is to present a design flow from specifications to layout for SI wave analogue filters, including a detailed description of the various optimisations employed. The proposed flow has three main steps. The first step (Section 2.2) involves the generation of the wave filter structure from a given filter frequency response specification and the optimization of wave coefficients. The second step (Section 2.3) involves eliminating block level distortion and reducing power consumption through a novel two stage bias and signal scaling approach. The third step (Section 2.4) focuses on the translation of the high level wave filter structure to a transistor level design, which includes first calculating initial dimensions then improving these through optimisation based on results from simulations. Matching in the physical layout is addressed, which is an essential consideration since the majority of the transistor level design consists of current mirrors, and their ratios.

2.1.3 Case study specifications

A case study is developed through this chapter in order to validate the methods and techniques described and to provide the basis for a prototype SI wave filter chip. The

specifications for this case study have been based on previous wave and SI filters presented in the literature. All previous fabricated SI wave filters [10, 11, 33] have focused on all-pole filters (Chebyshev), and so for the work in this chapter an elliptic response is chosen to demonstrate that other wave filter types using SI are possible. The presented filter is designed for a -3dB frequency of 100kHz which is higher than previously reported fabricated wave filters [11]. Values chosen for power consumption and SNR are comparable or lower than those obtained from integrator-based SI filter designs. For example the power consumption of the integrator based filter in [13] is 160mW. The case study specifications are shown in Table 2.1.

Table 2.1 Specifications for filter case study

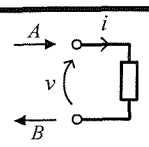
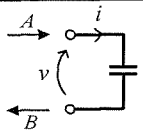
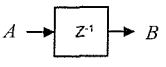
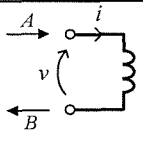
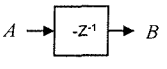
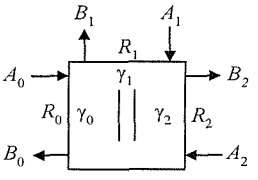
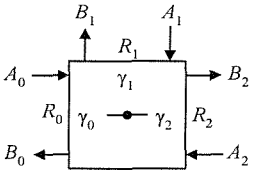
Specification:	Designed
Filter type:	3rd order lowpass elliptic
Passband ripple:	0.5dB
Stopband attenuation:	≥ 30dB
-3dB frequency:	100kHz maximum
Sampling frequency:	1MHz
SNR:	≥ 60dB
Supply voltage:	3.3V
Nominal bias current:	200μA
Power:	≤ 50mW
Technology:	0.6μm single poly CMOS

2.2 Ideal wave filter design flow

The ideal wave filter design flow consists of three steps. The first step (Section 2.2.1) involves the design of the passive prototype filter. The second step involves translating this passive prototype into a wave filter structure and calculating the wave filter's adaptor coefficients (Section 2.2.2). The last step is the optimisation of these wave filter coefficients (Section 2.2.3). Ideal circuit models for the wave filter circuit blocks are given in Section 2.2.4. Table 2.2 shows the constituent wave blocks and their models where it can be seen that a port resistance (or conductance, G) can be calculated from the equivalent passive component value, R, L or C, and the sampling period, T_s. Interconnection of passive components with different port resistances is

achieved through the use of adaptor blocks which are characterized by a set of three reflection coefficients γ_{ij}, where i is the adaptor number and j is the port number.

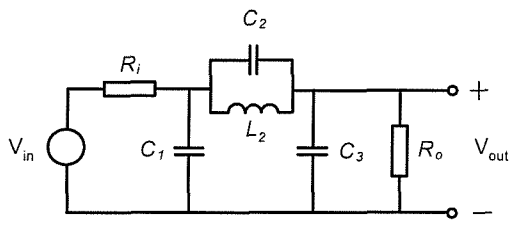
Table 2.2 Wave analogue filter blocks and their models

Circuit element	*Wave block*	*Resistance*	*Wave model*
 *Resistor*	*Implemented as adaptor block port resistance*	R	$A = 2i$ $B = 0$
 *Capacitor*	 *Positive delay cell*	$\frac{T_s}{2C}$	$B = z^{-1}A$
 *Inductor*	 *Negative delay cell*	$\frac{2L}{T_s}$	$B = -z^{-1}A$
Parallel connection of three circuit elements	 *Parallel Adaptor*	R_0, R_1, R_2	$\gamma_j = \frac{2G_j}{G_0 + G_1 + G_2}$ $G = \frac{1}{R}$ $A_N = \gamma_0 A_0 + \gamma_1 A_1 + \gamma_2 A_2$ $B_j = A_N - A_j,\ j = 0, 1, 2$
Series connection of three circuit elements	 *Series Adaptor*	R_0, R_1, R_2	$\gamma_j = \frac{2R_j}{R_0 + R_1 + R_2}$ $A_N = A_0 + A_1 + A_2$ $B_j = A_N - \gamma_j A_N,\ j = 0, 1, 2$

2.2.1 Prototype filter

The filter case study was designed to the specifications shown in Table 2.1. From suitable filter tables [98] it was found that the stopband attenuation and passband

ripple can be achieved using a 3rd order elliptic ladder filter, with structure and normalised values given in Figure 2.2.



$R_i = R_o = 1\ \Omega$
$C_1 = C_3 = 1.44483$ F
$C_2 = 0.20718$ F
$L_2 = 0.93666$ H

Figure 2.2 Ladder reference filter and normalised values

Wave analogue filters use the bilinear transformation between the continuous and discrete frequency domain and so it is necessary to pre-warp the response to compensate for the distortion which will be introduced by this transform. The pre-warped frequency ω_p can be found by:

$$\omega_p = \frac{2}{T_s}\tan\left(\frac{\omega_{co}T_s}{2}\right) \qquad (2.1)$$

where T_s is the sample period and ω_{co} is the desired cut-off frequency. Using the specified 1MHz sampling frequency gives a 1μs sample period and using Eq. (2.1) with the specified 100kHz (628.319krad/s) cut-off frequency gives ω_p of 649.839krad/sec. The normalised filter values are specified for a cut-off frequency defined at the -0.5dB level instead of -3dB, and this is now corrected for by dividing by the -3dB frequency which is found from simulation of the prototype filter to be 1.1379rad/s. The final ω_p is found as 571.0866krad/s and is now used to scale the normalised passive component values:

$$C_1 = C_3 = \frac{C'_1}{\omega_p} = 2.530\mu F \quad C_2 = \frac{C'_2}{\omega_p} = 0.363\mu F \quad L_2 = \frac{L_2}{\omega_p} = 1.640\mu H \qquad (2.2)$$

The resulting passive filter response is shown in Figure 2.3, which confirms that the cut-off frequency has been correctly placed at 649.8394krad/s (103.4251kHz) which should give a -3dB freq of 100kHz in the wave analogue filter.

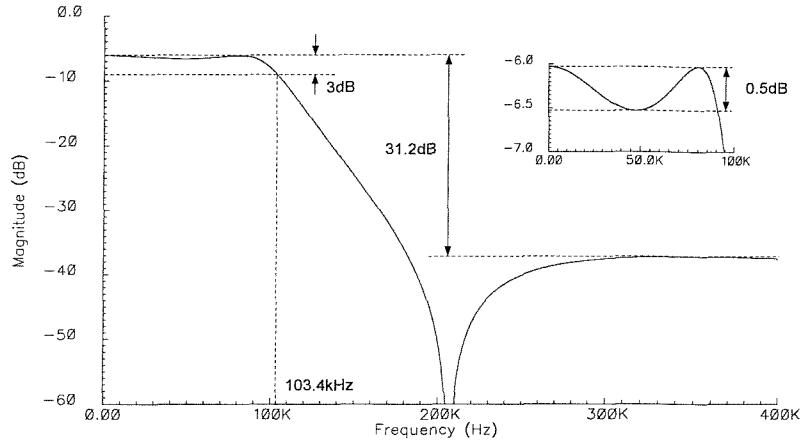


Figure 2.3 Response of the passive prototype filter

2.2.2 Wave architecture

Having designed the passive prototype filter, the wave analogue filter structure must now be determined and the adaptor coefficients which define the wave filter operation calculated and optimised. Using Table 2.2 and considering Figure 2.4, the wave filter is created by considering the passive filter as three parts, a parallel connected R_i and C_1, a parallel connected C_2 and L_2 and a parallel connected C_3 and R_o with these three parts connected in series. The correct wave structure, shown in Figure 2.4(b), consists of three parallel adaptors, one series adaptor, three positive delay cells and one negative delay cell. The operation of this wave filter is completely defined by the adaptor port coefficients, γ_{00} to γ_{32} which must now be calculated and optimised.

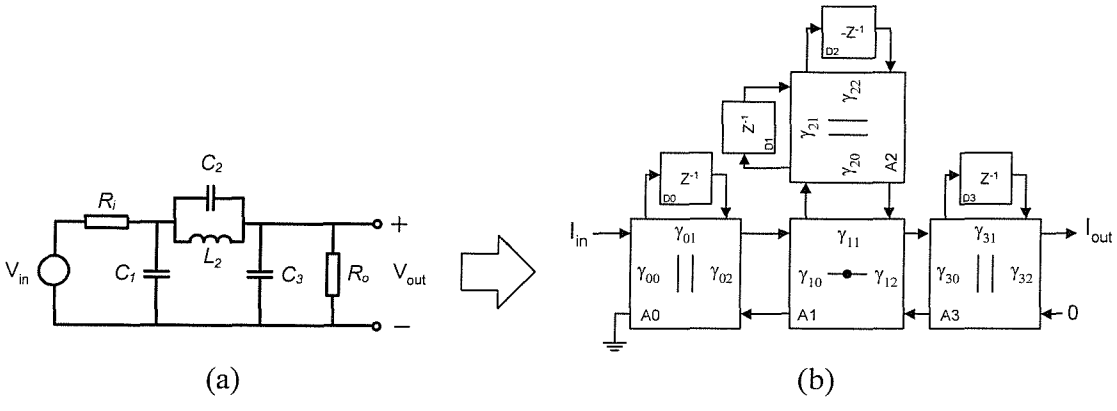


Figure 2.4 Passive reference filter (a) and wave equivalent structure (b)

First, all known port resistances are calculated, using Table 2.2:

$$R_{00} = R_{32} = 1.0 \tag{2.3}$$

$$R_{01} = \frac{T_s}{2C_1} = \frac{1\times10^{-6}}{2\times2.529967\cdot10^{-6}} = 0.197631 \tag{2.4}$$

$$R_{21} = \frac{T_s}{2C_2} = \frac{1\times10^{-6}}{2\times0.362782\cdot10^{-6}} = 1.378238 \tag{2.5}$$

$$R_{22} = \frac{2L_2}{T_s} = \frac{2\times1.640137\cdot10^{-6}}{1\times10^{-6}} = 3.280274 \tag{2.6}$$

$$R_{31} = \frac{T_s}{2C_3} = \frac{1\times10^{-6}}{2\times2.529967\cdot10^{-6}} = 0.197631 \tag{2.7}$$

$$R_{32} = R_o = 1.0 \tag{2.8}$$

Directly connected ports must have equal resistance, giving:

$$R_{02} = R_{10}\ ,\ R_{11} = R_{20}\ ,\ R_{12} = R_{30} \tag{2.9}$$

An important feature of wave analogue filters is the degree of freedom given due to these resistances not being completely constrained. This allows us to choose some of the coefficients, in this case three, to yield a more accurate filter realisation. As can be seen in Table 2.2, adaptor coefficients can be calculated from their port resistances using the following equations and constraint:

series adaptors: $$\gamma_j = \frac{2R_j}{R_0 + R_1 + R_2} \quad (2.10)$$

parallel adaptors: $$\gamma_j = \frac{2G_j}{G_0 + G_1 + G_2} \quad (2.11)$$

all adaptors $$\sum_{j=0}^{2} \gamma_{ij} = 2 \quad (2.12)$$

where i refers to the adaptor number and j refers to the port number. To begin with the following adaptor coefficients are arbitrarily set such that the entire filter coefficient collection is constrained:

$$\gamma_{02} = \gamma_{12} = \gamma_{20} = 1.0 \quad (2.13)$$

Rearranging the wave equations shown in Table 2.2 allows calculation of the port resistances of Eq. (2.9):

$$R_{02} = R_{10} = \frac{R_{00}R_{01}(2-\gamma_{02})}{\gamma_{02}(R_{00} + R_{01})} = 0.165 \quad (2.14)$$

$$R_{11} = R_{20} = \frac{R_{21}R_{22}(2-\gamma_{20})}{\gamma_{02}(R_{21} + R_{22})} = 0.970 \quad (2.15)$$

$$R_{12} = R_{30} = \frac{\gamma_{12}(R_{10} + R_{11})}{2-\gamma_{12}} = 1.135 \quad (2.16)$$

With a complete set of port resistances it is a simple matter to calculate the remaining adaptor coefficients, using Eq. (2.10) to (2.12), leading to the initial coefficient set of Table 2.3.

Table 2.3 Initial filter coefficients

Coefficient	Initial value	Coefficient	Initial value
γ_{00}	0.165018	γ_{20}	1.0
γ_{01}	0.834982	γ_{21}	0.704147
γ_{02}	1.0	γ_{22}	0.295853
γ_{10}	0.145327	γ_{30}	0.253773
γ_{11}	0.854673	γ_{31}	1.458068
γ_{12}	1.0	γ_{32}	0.288159

2.2.3 Wave coefficient optimisation

In practice, the performance of wave analogue filters will be limited by a number of non-ideal factors associated with real transistor level design. When designing the wave filter, the coefficient calculation stage has degrees of freedom allowing a number of coefficients to be arbitrarily chosen (Eq. (2.13)). Many equivalent coefficient sets will therefore give the same filter response, and this can be exploited to obtain the most suitable set of coefficients from a circuit implementation perspective. The techniques given in [10] have been adapted to optimise for two factors, which are now described:

Coefficient spread: The adaptor coefficients γ_{ij} (where i is the adaptor number and j is the port number) at circuit level are implemented using current mirror branch ratios. Smaller coefficients will result in a current mirror branch with smaller transistors, which are subject to greater sensitivity to matching and process variations than large transistors. Since all coefficients in any one adaptor must sum to two (Eq. (2.12)), then sensitive designs can be avoided by ensuring that the spread of coefficient values is as small as possible. Eq. (2.17) provides a sensitivity cost, E_s, which increases as the coefficients, γ_{ij}, decrease:

$$E_s = \frac{1}{\gamma_{ij}} \qquad (2.17)$$

Technology grid: All transistor widths and lengths in the final design must lie on a standard working grid (a process parameter, for this case study 0.05μm) and so rounding errors can exist when implementing the wave filter design. Since many current mirror branch widths will depend on adaptor coefficients, careful choice of the free coefficients can ensure that the number of widths which lie closer to the technology grid are maximised. A cost calculation for the percentage error, E_t, incurred when implementing the widths can now be found from the calculated width W' and the technology grid resolution, λ:

$$E_t = \left| \frac{\lambda}{W'} round\left(\frac{W'}{\lambda} \right) - 1 \right| \times 100 \qquad (2.18)$$

It has been found through simulation that filters with un-optimised wave adaptor coefficients have inferior performance in terms of passband attenuation and cutoff frequency accuracy when compared to filters with optimised coefficients. The degradation in performance depends on the filter type and order, but from extensive analyses of numerous designs, a cut-off frequency error of as much as 6% in the case of 3rd order filters and up to 9% in the case of 5th order elliptic filters can result from un-optimised coefficient use. By automating the coefficient calculations it is possible to quickly determine new coefficient sets from particular choices of free coefficients. This allows the free coefficient space to be searched for values which give a smaller E_s and E_t. For example, the following pseudo code listing shows an algorithm which traverses the free coefficient (`FreeCoeffs`) space, and at each point calculates the remaining coefficients using a function, `CalculateCoeffs()`.A sensitivity matrix (`SensMatrix`) and technology grid matrix (`TechMatrix`) is generated for each adaptor and a corresponding E_s and E_t cost is calculated based on the worst sensitivity or technology grid rounding errors. These are then combined to give a total cost (`TotalCost`) and if the combined cost is better than the current best cost (`BestCost`) then the current coefficient set are considered optimal (`OptCoeffs`). Using this method the optimised filter coefficients are given in Table 2.4. As it can be seen the initial smallest coefficient is 0.145, whilst the optimised smallest coefficient is 0.279, leading to a less sensitive design.

```
traverse FreeCoeffs space {
        CalculateCoeffs(FreeCoeffsChoice)
                calculate SensMatrix for each adaptor
                        generate Es cost based on worst sensitivity
                calculate TechMatrix for each adaptor
                        generate Et cost based on worst grid errors
                combine Et and Es to give TotalCost
        }
        if TotalCost is better than BestCost cost then {
                BestCost = TotalCost
                OptCoeffs = Coeffs
        }
}
```

Table 2.4 Optimised filter coefficients

Coefficient	Initial value	Coefficient	Initial value
γ_{00}	0.279262	γ_{20}	0.615385
γ_{01}	1.413046	γ_{21}	0.974972
γ_{02}	0.307692	γ_{22}	0.409643
γ_{10}	0.451707	γ_{30}	0.302128
γ_{11}	1.086745	γ_{31}	1.417692
γ_{12}	0.461538	γ_{32}	0.280180

2.2.4 Ideal simulations

Before progressing onto the full transistor level design of the wave filter, it is important to verify the filter response at ideal wave level, and hence determine that a suitable set of coefficients and wave structure has been calculated. Because SI wave filters are sampled networks, a conventional AC simulator cannot be used to derive the frequency response since it requires the circuit to be linear. SpectreRF performs a *periodic* steady state analysis to linearise the switched circuit about its sampling frequency and then uses this information to perform a *periodic* AC analysis which gives the same type of data as the normal AC analysis of a linear circuit [99]. An ideal circuit equivalent for a three output current mirror is shown in Figure 2.5 where the parameters B_1, B_2 and B_3 determine the current mirror branch ratios. An ideal delay cell can be seen in Figure 2.6 which models the first generation delay cell, requiring the two non-overlapping clocks ϕ_1 and ϕ_2 derived from the sampling

frequency f_s such that the output current sample is delayed by one clock period ($1/f_s$). These models were used to create the ideal circuit for the case study filter, and simulating with SpectreRF gives the ideal wave filter response of Figure 2.7. As can be seen, the response has a -3dB frequency of exactly 100kHz as expected.

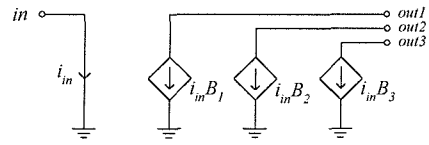


Figure 2.5 Ideal circuit for current mirrors

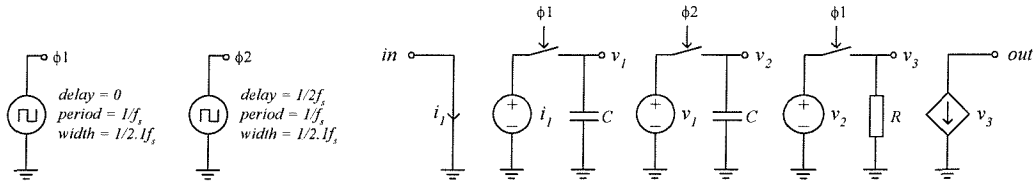


Figure 2.6 Ideal circuit for delay cell

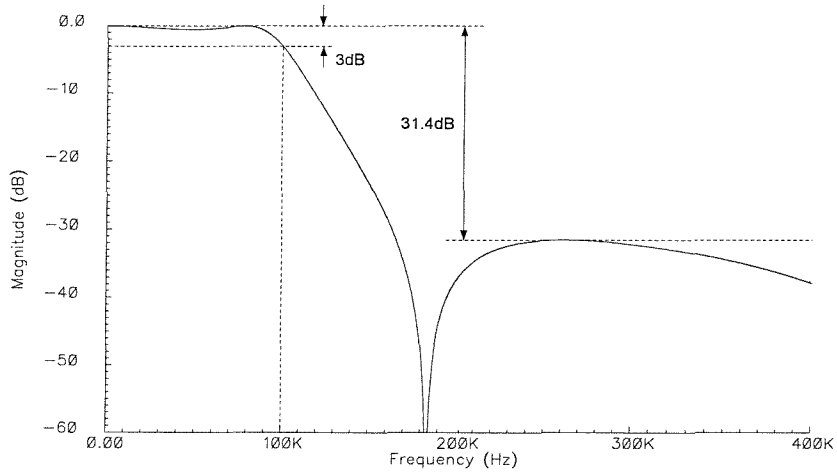


Figure 2.7 Ideal wave filter response

2.3 Power aware scaling method

In order to investigate the power consumed in the designed filter, transistor level implementations of the wave blocks are now considered. Table 2.6 (Page 45) shows the circuit level structures used to implement series and parallel adaptors [53] and positive and negative delay cells, the latter consisting of the high accuracy S^2I

memory cell [100]. The total quiescent power consumption of a wave filter is shown in Eq. (2.19), which is derived by multiplying the supply voltage, V_{dd}, by the sum of all the bias currents, J, in the wave filter. The bias currents contributions for each of the four wave blocks can be determined from Table 2.2, and these are then multiplied by the number of occurrences of this block in the wave filter, where S and P are the total number of series and parallel adaptors and D_p & D_n are the total number of positive and negative delay cells. Note that in the adaptor blocks, the three branches implementing adaptor coefficients contribute only $2J$, due to Eq. (2.12), and hence each series or parallel adaptor contributes a total of $18J$ or $12J$. respectively, which is one less than the number of branches in their circuits.

$$Power = JV_{dd}\left(18S + 12P + 2D_p + 4D_n\right) \quad (2.19)$$

Section 2.3.1 investigates the spread of signal levels in wave filters and how this is addressed with existing techniques. Section 2.3.2 then details the proposed power-aware two stage bias and signal scaling method. Section 2.3.3 considers noise in the context of the proposed approach. Finally, Section 2.3.4 considers how to integrate the proposed method into the design flow for wave filters.

2.3.1 Non-homogenous signal levels in wave filters

Where internal signal levels within a large design vary significantly, these are referred to as non-homogeneous [11]. Unfortunately, non-homogeneous signal levels within wave filters can lead to the dynamic range of some blocks being exceeded, and it is essential to investigate such possible sources of distortion. The nature of wave designs makes it difficult to derive mathematical relationships between the input signal and an arbitrary internal node and consequently it is far more practical to model the ideal circuit and use SpectreRF to derive the frequency response. Extensive libraries of behavioural and ideal models have been developed for this purpose [38]. To illustrate block level distortion, a 5^{th} order Chebyshev SI wave filter was designed and is shown in Figure 2.8, with two internal node responses. It is clear from these responses that some nodes exhibit a substantial amount of gain from the input, and this can result in the dynamic range of that current mirror being exceeded.

Without addressing this issue the performance of the filter would be unlikely to meet its expected values. In a Class A design there are two ways to deal with this problem, either increase the bias current for the current mirrors where the signal peaks are large, or decrease the signal values. Previous wave filter publications appear to have simply identified those blocks whose dynamic range are being exceeded and introduced circuitry and scaling to prevent distortion at these points only [11, 12].

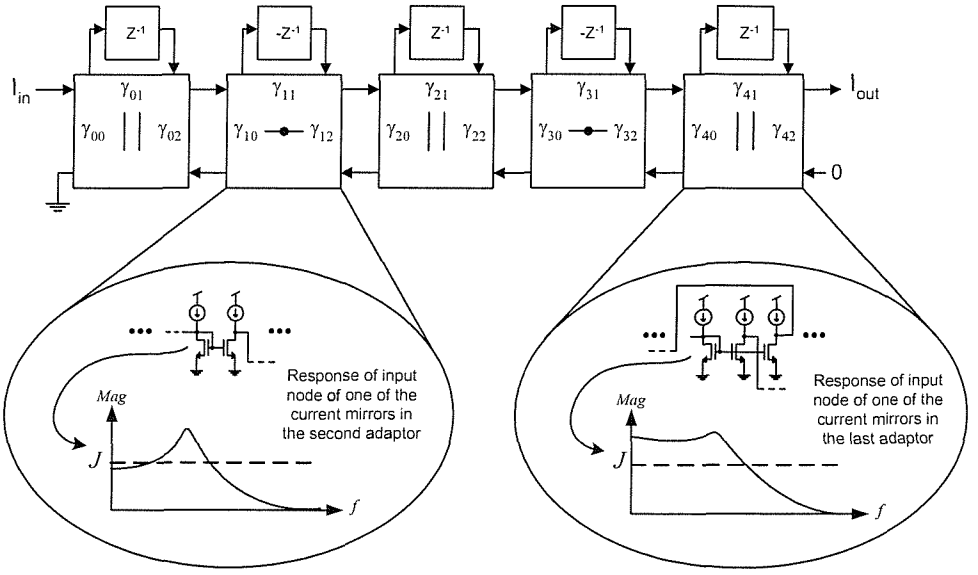


Figure 2.8 Example filter with two internal node responses shown

2.3.2 Two step bias and signal scaling method

The proposed power-aware scaling method consists of two stages. Stage 1 involves scaling current mirror *bias* levels, and stage 2 involving scaling delay cell *signal* levels. Both of these changes can be made independently of one another and without affecting the rest of the filter. The procedure starts after the wave filter coefficients have been calculated and the first task is to determine the peak input signal level, i_{peak} to all delay cells and adaptor current mirrors. Behavioural or ideal models of the wave blocks allow an ideal periodic AC analysis of the circuit to be run using SpectreRF and if a unity input is chosen for these simulations then the peak magnitudes on the frequency response are in fact the peak input signal levels, i_{peak} . The two stages are now discussed in detail.

Stage 1: ***Current mirror bias scaling:*** As shown in Figure 2.9, every current mirror block in the adaptors has its bias current, J, scaled to a new value, J', which just supports the peak input signal value, i_{peak}, found from the frequency response for that block. Not only does this prevent distortion in this block, but it also lowers power consumption by scaling down bias levels where the signals they support are small. The bias currents in all current mirrors can be easily altered by scaling all the transistor widths. Note that the branch ratios are not changed in stage 1, just the bias current and hence the maximum supported signal current. A scaling factor, S, is calculated using Eq. (2.20), then transistor widths in the current mirror are scaled according to Eq. (2.21).

$$S = \frac{i_{peak}}{i_{filter}} \qquad (2.20)$$

$$W_{new} = S.W_{old} \qquad (2.21)$$

where i_{peak} is the peak input signal value to the current mirror, i_{filter} is the AC input current to the entire filter and W_{old} and W_{new} are the old and new width dimensions for the current mirror being scaled. Note that the scaling in this stage is *relative* to the initial peak input value, and for that reason the designed maximum signal to bias current modulation level is always respected. The scaling process therefore does not compromise the linearity of the current mirrors.

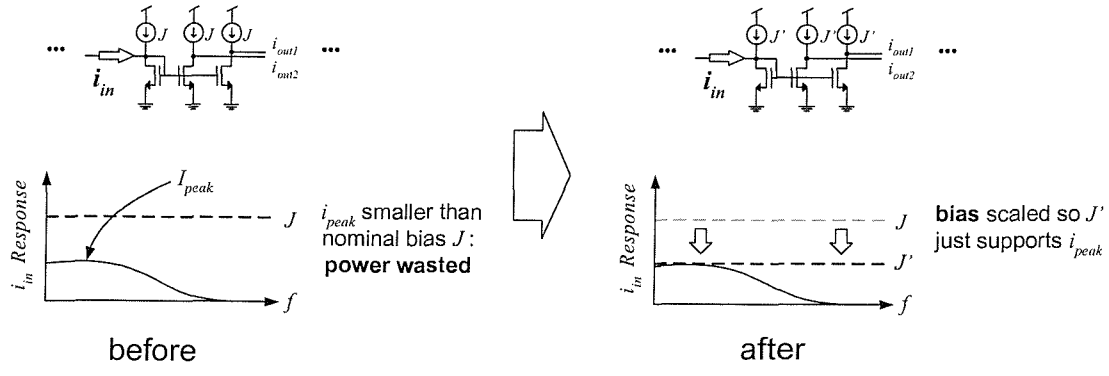


Figure 2.9 Bias scaling in Stage 1, before and after proposed method

Stage 2: Delay cell signal scaling: Unlike the current mirror bias scaling in stage 1, the delay cell bias current cannot be changed without significantly affecting the performance of the cell, so a different approach is taken in stage 2. Instead, as shown in Figure 2.10 the signal values in the delay cells are scaled such that the peak input signal, i_{peak}, through these will always be the designed maximum value, and hence just supported by the nominal bias current J. No extra scaling blocks are required to achieve this and instead the new dimensions are implemented in any output branches of current mirrors connected to the input of the delay cell and any input branches of current mirrors connected to the output of the delay cell. The scaling factor, S is again calculated using Eq. (2.20) and the necessary branches then have their transistor widths altered as follows:

$$W_{new} = \frac{W_{old}}{S} \qquad (2.22)$$

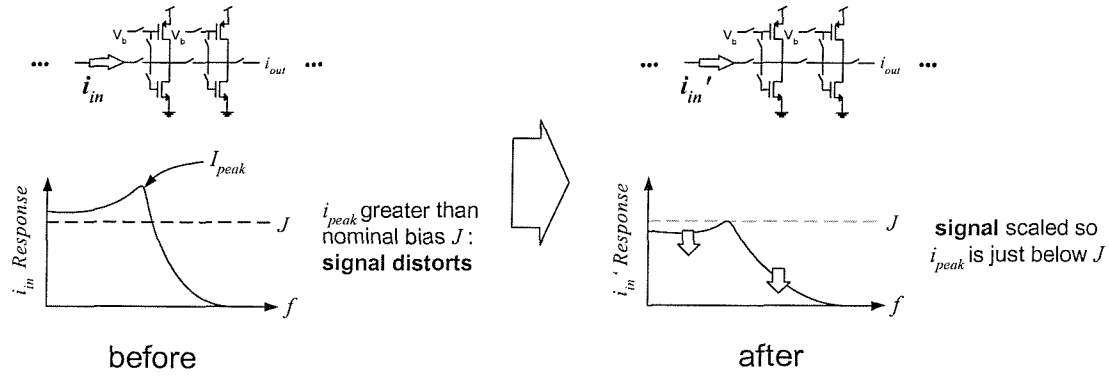


Figure 2.10 Signal scaling in Stage 2, before and after proposed method

These changes must not affect the rest of the filter and as such the signal levels everywhere other than the delay cell and adjoining current mirror branches should remain the same. An example situation is shown in Figure 2.11 and Eq. (2.23) shows the current mirror and delay cell ratios, α_n and α_D respectively. It is clear from Eq. (2.24) that if these are multiplied from the first altered mirror to the last then the result is unity, meaning the scaling in stage 2 does not affect the rest of the filter. Furthermore, similar to stage 1, the scaling in stage 2 is a *relative* change and as such the designed modulation level of the delay cell is always respected.

$$\alpha_1 = \alpha_2 = \frac{1/S}{1} = \frac{1}{S} \qquad \alpha_D = 1 \qquad \alpha_3 = \frac{1}{1/S} = S \tag{2.23}$$

$$\alpha_{tot} = \alpha_1.\alpha_D.\alpha_3 = \frac{1}{S}.1.S = 1 \tag{2.24}$$

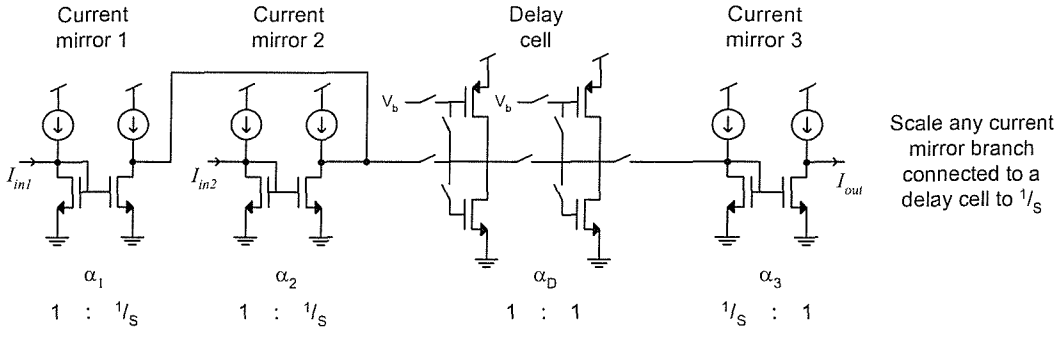


Figure 2.11 Scaling of current mirrors either side of delay cells

By appropriately assigning bias currents and signal levels, stage 1 and stage 2 ensure that signal levels never exceed their biasing. Furthermore, unlike methods where a uniform bias current is assumed, the power consumption of the filter is reduced by lowering bias currents where possible. Note that it is only the current mirror branches in the adaptors which are scaled at all, the delay cells are left as they are, thus ensuring correct memory cell operation.

2.3.3 Noise considerations

In the context of Class A SI wave filters, the two main sources of noise are thermal and 1/f, which are the first and second terms in Eq. (2.25) respectively [101].

$$\overline{i_d^{\,2}} = 4kT\left(\frac{2}{3}g_m\right)\Delta f + K\frac{I_D}{f}\Delta f \tag{2.25}$$

where $\overline{i_d^{\,2}}$ is the small signal noise, k is Boltzmann's constant, T is temperature, Δf is bandwidth, K is a constant for a given device, and f the frequency. The thermal

noise term includes transconductance, g_m as a factor, which is proportional to the device drain current, I_D. The second term includes I_D explicitly, and as such the total small signal noise is proportional to the quiescent bias level of the transistor. Therefore, at the block level if bias currents are increased, large signals will be less distorted but noise will increase, leading to a trade off between large signal linearity and small-signal noise. However, the proposed methodology only increases bias currents where absolutely necessary, and in fact the power savings achieved imply that the filter's average bias current is reduced, and hence the noise should also be lower. In terms of the entire filter, the proposed method therefore improves large signal linearity by scaling signals and bias currents where necessary but due to the general reduction in power, and hence average bias current, the small signal noise should be reduced.

2.3.4 Power aware SI wave filter design flow

A power-aware design flow for switched-current wave filters incorporating the proposed method is now given. Once the basic wave structure has been designed [12], scaling factors have to be calculated for every branch of every current mirror in each adaptor block. There are three components to the final scaling factor, the first represents the alterations made during stage 1, the second implements the changes made during stage 2 and the last implements the calculated adaptor coefficients, γ_{ij}. These three components are represented with multi-dimensional arrays, and the index of the array elements uniquely identifies every current mirror branch: i is the adaptor number, j the current mirror number in the adaptor and k the branch in that current mirror. These scaling components are now discussed in detail:

A_{ijk} – Peak signal values are read from the magnitude response of the input of all the current mirrors as described in stage 1 of Section 2.3.2. These values are practically obtained by connecting an AC source of value unity to the input of the wave filter and using a switched-mode simulator to plot the response of the current mirror inputs. This array, A_{ijk}, is then filled with the resulting scaling factors as found from Eq. (2.20).

$\boldsymbol{D_{ijk}}$ – Peak signal values for the delay cells are obtained in exactly the same manner as for the current mirrors. The structure of the filter is then considered and all current mirror branches, be they outputs or inputs, which are connected to the input or output of a delay cell are identified. These positions in D_{ijk} are then filled with the inverse of the delay cell scale value as described in stage 2 of section 2.3.2. The rest of the array is unity filled.

$\boldsymbol{Y_{ijk}}$ – The adaptor coefficient values, γ_{ij} are implemented in the correct positions of this array, Y_{ijk}, reflecting the relevant branches of the current mirrors in the adaptor blocks of Table 2.2. The rest of the array is unity filled.

The entire set of scaling factors S_{ijk} encompassing bias and signal scaling and wave coefficients are then given by Eq. (2.26) as the product of the contributions from the above arrays. Given a set of nominal current mirror dimensions, the new widths for all the branches in the entire wave filter can be calculated from Eq. (2.27).

$$S_{ijk} = A_{ijk} D_{ijk} Y_{ijk} \tag{2.26}$$

$$W_{new,ijk} = S_{ijk} W_{old,ijk} \tag{2.27}$$

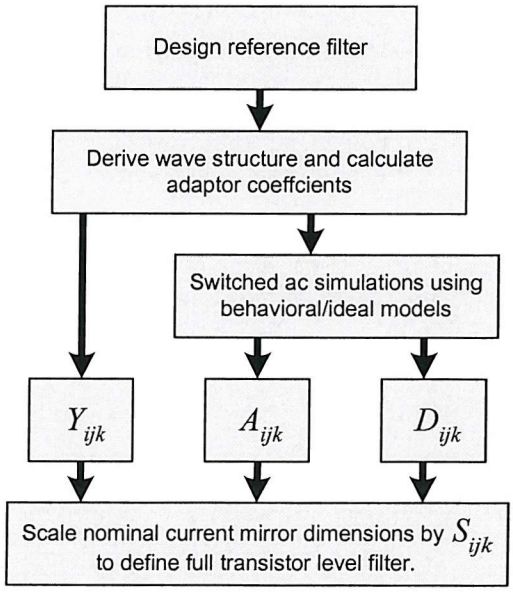


Figure 2.12 Power aware SI design flow

The resulting power-aware switched-current wave filter design flow is shown in Figure 2.12. The in-house tool developed during the course of this research, AutoSIF [38], has recently had its functionality extended to incorporate the power-aware steps. The tool is written in SKILL®, is integrated into Cadence Design Framework II and is the focus of the next chapter. In order to illustrate the two stage scaling process, the scaling procedure for the first adaptor in the case study elliptic 3rd order filter is numerically detailed in Table 2.5. Using simulation with ideal models, the current mirror peak values and the delay cell peak value were found and using the coefficients for the first adaptor, the total scaled values are shown in the last column.

Table 2.5 Scaling factors for the first adaptor

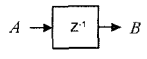
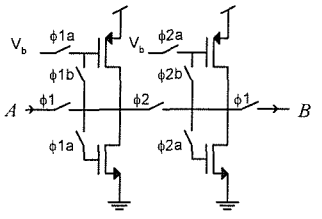
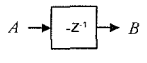
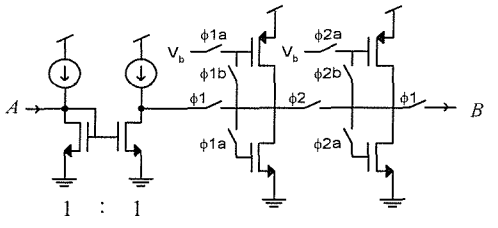
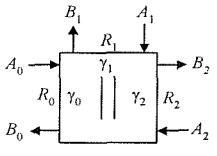
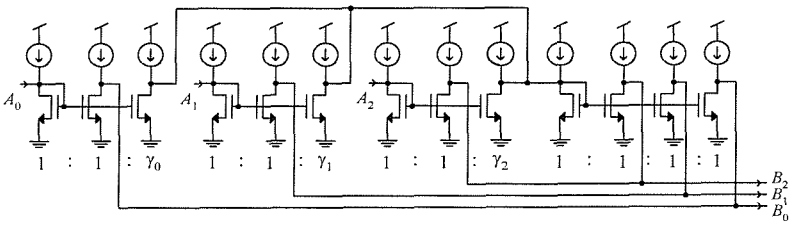
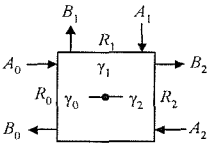
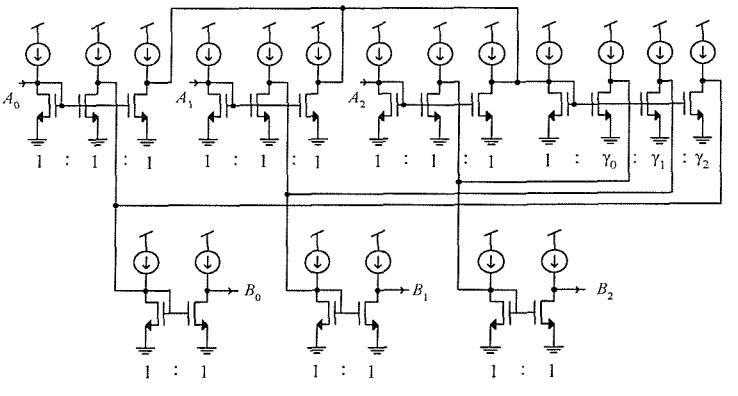
Branch adaptor mirror branch ijk	Peak value	A scale	D scale	Y scale	Total A×D×Y
{Delay cell value found to be 0.7051}					
0,0,0	0.9701	0.9701	1	1	0.9701
0,0,1	-	0.9701	1.4182	1	1.3758
0,0,2	-	0.9701	1.4182	0.279262	0.3842
0,1,0	0.7905	0.7905	1	1	0.7905
0,1,1	-	0.7905	1	1	0.7905
0,1,2	-	0.7905	1	1.413046	1.11170
0,2,0	0.7792	0.7792	1	1	0.7792
0,2,1	-	0.7792	1	1	0.7792
0,2,2	-	0.7792	1	0.307692	0.2398
0,3,0	1.3406	1.3406	1	1	1.3406
0,3,1	-	1.3406	1	1	1.3406
0,3,2	-	1.3406	1	1	1.3406
0,3,3	-	1.3406	1	1	1.3406

2.4 Transistor level design

Having produced the filter block diagram (Figure 2.4), generated the optimised adaptor coefficients (Table 2.4), and employed the two part power aware scaling strategy in Section 2.3, the next step is to produce the transistor level design. Table 2.6 shows the transistor circuitry of the wave blocks. The positive and negative delay cells require two SI memory cells and the parallel and series adaptor blocks consist

of current mirrors with some branch ratios given by the adaptor coefficients. The complete schematic for the filter is shown in Figure 2.13. To produce the transistor dimensions of the delay cell and the adaptor blocks, analytical expressions relating filter performance specifications to transistor dimensions are now developed.

Table 2.6 Transistor implementations of wave filter blocks

Wave block	Transistor implementation
 *Positive delay cell*	
 *Negative delay cell*	
 *Parallel Adaptor*	
 *Series Adaptor*	

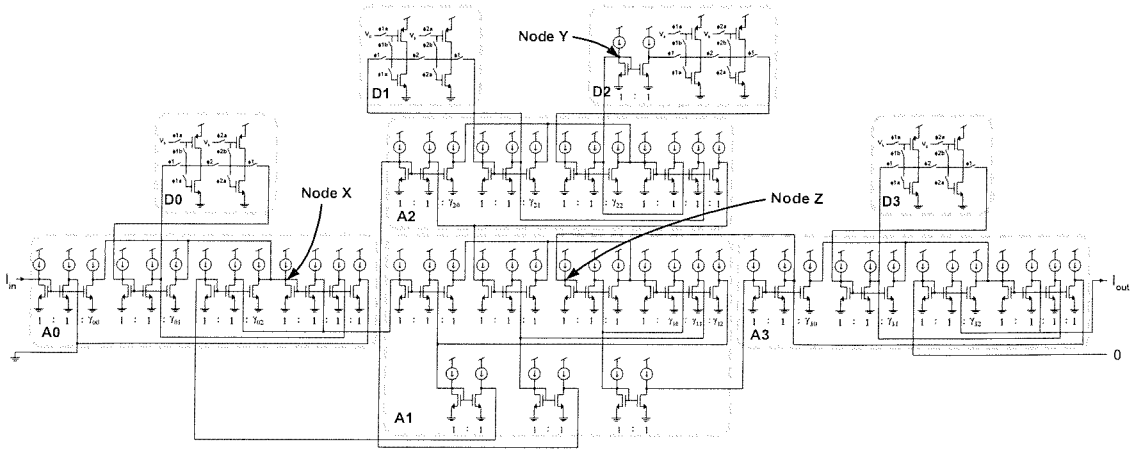


Figure 2.13 Complete transistor level filter

The delay cell design is considered in detail in Section 2.4.1 and the current mirror design for the adaptor blocks is detailed in Section 2.4.2. Periphery bias and clock generation blocks are detailed in Sections 2.4.3 and 2.4.4. A strategy for layout of matched transistor stacks, which can be used for all of the current mirrors, delay cells and biasing structures is shown in Section 2.4.5.

2.4.1 Delay cell

Numerous SI memory cell designs have been reported in the literature, and the S^2I memory cell [6] has been chosen because at the time of this work, its capability at reducing charge injection and conductance ratio errors represented the state of the art [6]. Furthermore, this is the first time the S^2I memory cell has been used in a wave filter prototype chip, since all fabricated wave filters in the literature have employed the older first generation memory cell [11, 12]. The S^2I cell is shown in Figure 2.14, and the first phase, ϕ_1, is split into two subphases, ϕ_{1a} and ϕ_{1b}. During phase ϕ_{1a} the circuit is configured as a normal second generation memory cell and a 'coarse' current memory is stored on the NMOS 'coarse' transistor whilst the PMOS 'fine' transistor is configured as a current source. During phase ϕ_{1b}, the PMOS 'fine' transistor is diode connected and since the input current is still flowing, this transistor sources the bias current and the errors from the coarse phase. On the output stage, ϕ_2, the output current contains no errors from the coarse phase, just the errors from the fine stage, which are significantly smaller [6].

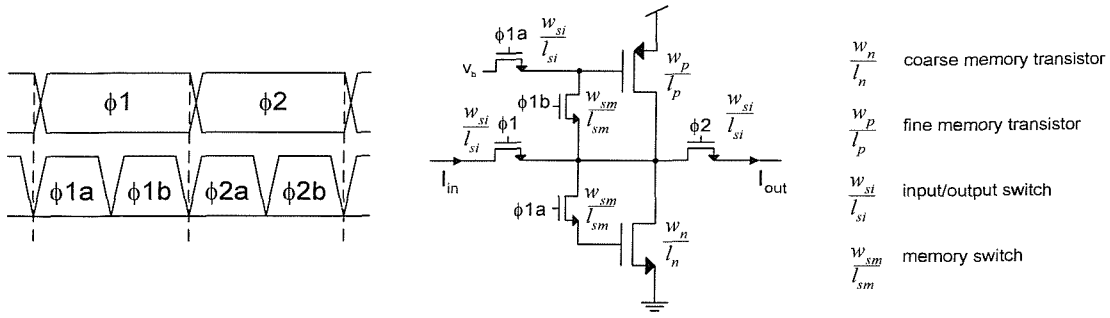


Figure 2.14 S^2I memory cell parameters and clock phases

Initial design: : The design of an SI delay cell requires two S^2I memory cells (Figure 2.14) and requires derivation of the transistor dimensions, w_p, l_p, w_n, l_n, w_{si}, l_{si}, w_{sm} and l_{sm}. A delay cell dimensioning method has been developed, adapted from [14], which uses analytical expressions based on square-law saturated models to calculate an initial design. Since square-law models have limited accuracy when compared to real MOS behaviour, this accuracy is improved by optimising this initial design based on simulation results. The complete delay cell design process is shown in Figure 2.15 where user specifications and required memory cell parameters are shown on the left and process parameters on the right. The starting point in the design process is to find out the filter bias current for a given power consumption P_{tot}, which is the first user input to the procedure shown in Figure 2.15.

$$J = \frac{P_{tot}}{V_{dd}\left(18S + 12P + 2D_p + 4D_n\right)} \qquad (2.28)$$

where S and P are the total number of series and parallel adaptors, D_p and D_n are the total number of delay cells used, J is the bias current and V_{dd} is the supply voltage. Eq. (2.28) is obtained by summing all the quiescent current contributions from each wave block. From Table 2.6 it is noted that the adaptor blocks contribute one bias current less than expected due to Eq. (2.12).

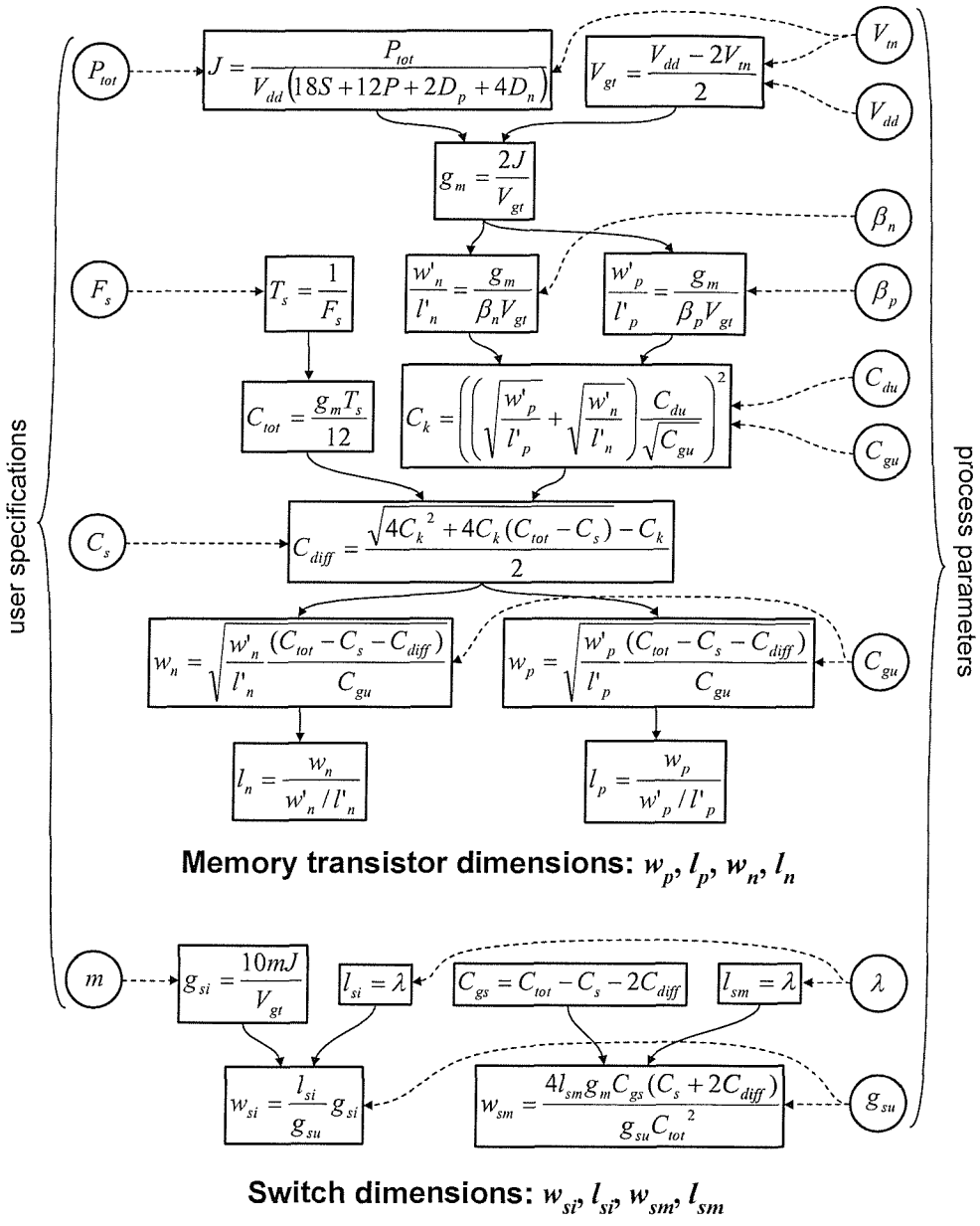


Figure 2.15 S^2I memory cell calculations

From Table 2.1 the designed filter power consumption is 50mW and allowing an estimate of 5mW for the clock and bias structures leaves 45mW for the filter. With a power supply of 3.3V, this gives a maximum current of 213μA; a smaller bias current J of 200μA is chosen to allow a margin for error in the power estimation for the clock and bias structures, giving a total estimated power of 47.24mW. Assuming the NMOS and PMOS threshold voltages V_{tn} and V_{tp} are the same (but opposite in polarity), it is possible to calculate the suggested gate overdrive voltage V_{gt}, as 0.8V [14]:

$$V_{gt} = \frac{V_{dd} - 2V_{tn}}{2} \quad (2.29)$$

Now using V_{gt} and the chosen bias current, J, the transconductance of the fine or coarse memory cell transistor, g_m [14] can be calculated:

$$g_m = \frac{2J}{V_{gt}} \quad (2.30)$$

This gives a g_m of 500μS. During the sampling phases, the time constant, τ, of either of the diode connected fine or coarse memory transistors is given by:

$$\tau = \frac{C_{tot}}{g_m} \quad (2.31)$$

where C_{tot} is the total capacitance at the summing node and g_m is the transconductance from Eq. (2.30). To make the fine and coarse settling times equal, C_{tot} is designed to be equal on both phases. Settling in the S^2I cell continues from the coarse to the fine phase and so half the sample period is available for settling. If an equivalent of greater than 8 bits settling accuracy is required, the memory cell drain current needs to settle to within at least 2^{-8} = 0.391% of the final value during the sample period, which requires 6 time constants (e^{-6} = 0.248%). The maximum sampling frequency, F_s, which is the second user input to the procedure (Figure 2.15), can now be used to calculate T_s and then C_{tot} using:

$$C_{tot} = \frac{g_m T_s}{12} \quad (2.32)$$

The specified sampling frequency is 1MHz (Table 2.1), giving T_s as 1μs and for the case study the memory cell is designed for a sampling rate of up to five times this value, allowing it to cope with the load of a second delay cell and current mirrors.

This gives C_{tot} as 8.333pF. Also, from square law saturated equations the ratio of w'/l' for the memory transistors can be found:

$$\frac{w'_n}{l'_n} = \frac{g_m}{\beta_n V_{gt}} \qquad \frac{w'_p}{l'_p} = \frac{g_m}{\beta_p V_{gt}} \tag{2.33}$$

The technology used to fabricate the filter is a CMOS 0.6μm process (Table 2.1) and the values for β_n and β_p are 77μA/V^2 and 25μA/V^2 respectively. Using these and the values from Eq. (2.29) and Eq. (2.30), it is found that w'_n/l'_n is 8.117 and w'_p/l'_p is 25.0. The total node capacitance, C_{tot}, mainly consists of the fine and coarse memory capacitances, however, the depletion capacitance formed from the drain diffusion area and the transistor bulk (connected to the source) from both the fine and coarse transistors should also be taken into account as should the memory switch depletion capacitance. The latter is normally estimated as a lumped element with the interconnect strays, called C_s which is an input to the procedure (Figure 2.15). To calculate w_n and w_p a quadratic in C_{diff} needs to be solved, which is simplified through an expression for C_k [14]:

$$C_{diff} = \frac{\sqrt{4C_k{}^2 + 4C_k(C_{tot} - C_s)} - C_k}{2} \tag{2.34}$$

$$C_k = \left(\left(\sqrt{\frac{w'_p}{l'_p}} + \sqrt{\frac{w'_n}{l'_n}} \right) \frac{C_{du}}{\sqrt{C_{gu}}} \right)^2 \tag{2.35}$$

where C_{gu} is the capacitance per unit area for the gate regions, and C_{du} a capacitance per unit length for the drain diffusion regions. With a C_{gu} of 2.1fF/μm and a C_{du} of 0.34fF/μm^2 C_k is calculated as 3.39fF. Assuming C_s as 15% of C_{tot} gives C_{diff} as 153.33fF. Now using Eq. (2.36), the calculated NMOS and PMOS transistor widths are 161.84μm and 284.03μm respectively, after which the lengths can be calculated from Eq. (2.37) giving the memory transistor sizes (rounded to the technology grid) as: NMOS 161.85/19.95 and PMOS 284.05/11.35.

$$w_n = \sqrt{\frac{w'_n}{l'_n}\frac{(C_{tot} - C_s - 2C_{diff})}{C_{gu}}} \qquad w_p = \sqrt{\frac{w'_p}{l'_p}\frac{(C_{tot} - C_s - 2C_{diff})}{C_{gu}}} \tag{2.36}$$

$$l_n = \frac{w_n}{w'_n / l'_n} \qquad l_p = \frac{w_p}{w'_p / l'_p} \tag{2.37}$$

This completes the design stage of determining the initial fine and coarse transistor dimensions of the memory cell. All that remains is to derive initial dimensions for the input/output and memory switches (Figure 2.14). CMOS switches are often used in SC circuits, however, these require an additional inverted clock signal for every clock phase, which adds complexity to the clock generation circuitry. Furthermore, using a CMOS switch in a memory cell settling loop can complicate cell settling. For these reasons, NMOS switches are used for the input/output and memory switches in the delay cell, since their simplicity trades off well against their lower conductance. Two factors must be considered for the input/output switches; switch on-resistance, which requires a high aspect ratio, and depletion capacitance which requires a small area. Both requirements indicate that l_{si} should be set to the minimum allowable gate length (in our process 0.6μ). To decide w_{si}, a rule-of-thumb is applied as to how much voltage drop is permissible on the switch, and it is suggested in [14] from experience that 10% of the designed V_{gt} gives good results. Using the modulation level m, (maximum signal to bias current ratio) which is the last input to the procedure of Figure 2.15, gives:

$$g_{si} = \frac{10mJ}{V_{gt}} \tag{2.38}$$

$$w_{si} = \frac{l_{si}}{g_{su}} g_{si} \tag{2.39}$$

where g_{si} is the allowed switch conductance, g_{su} is the unit switch on-conductance (a process related parameter) and w_{si} and l_{si} are the input/output switch dimensions. With g_{su} of 100μS and modulation index, m, as 0.75 (a typical value) this gives a width of 11.25μm and hence input/output switch dimensions of 11.25/0.6. The memory switch length must be made as short as possible to keep charge injection problems to a minimum, this gives 0.6μm (the minimum dimension in a 0.6μm process). This switch conductance forms a second order system given by Eq. (2.40) with the memory transistor's gate and depletion capacitances, and so Eq. (2.41) is used in order to be critically damped and settle as fast as possible. Using this equation gives a width of 1.822μm leading to the initial memory switch dimensions of 1.8/0.6. The complete set of initial transistor dimensions is given in Table 2.7.

$$Q = \frac{\sqrt{\frac{g_m}{g_{sm}} C_{gs}\left(C_s + 2C_{diff}\right)}}{C_{tot}} \qquad C_{gs} = C_{tot} - C_s - 2C_{diff} \tag{2.40}$$

$$w_{sm} = \frac{4 l_{sm} g_m C_{gs} (C_s - 2C_{diff})}{g_{su} C_{tot}^{\ 2}} \tag{2.41}$$

Table 2.7 Initial and improved S^2I memory cell dimensions

Transistor	Initial (μm)	Improved (μm)
PMOS fine memory w_p/l_p	161.85/19.95	150.3/20.95
NMOS coarse memory w_n/l_n	284.05/11.35	272.65/12.35
Input/Output switch w_{si}/l_{si}	11.25/0.6	11.25/0.6
Memory switch w_{sm}/l_{sm}	1.8/0.6	1.8/0.6

Simulation based improvement: The initial transistor dimensions have been obtained up to this point using the assumption of square law saturated behaviour, which would give some inaccuracy. While it may be possible to modify the analytical expressions to increase this accuracy, it is easier and more accurate to use optimisation based on accurate simulation results and this approach is now detailed. The simulated drain current and gate capacitance can be found using a DC simulation of the diode

connected coarse (NMOS) and fine (PMOS) memory transistors, as shown in Figure 2.16(a). The time domain settling response of the memory cell can be determined by hard-wiring it in its different phases and applying a step change in input current as shown in Figure 2.16(b). The fine and coarse gate capacitances, C_{gsp} and C_{gsn}, are normally designed to be equal yet both these and the drain current, I_D are affected by altering the memory cell transistor dimensions, w_p/l_p and w_n/l_n and so it is important to satisfy either of the adjustments without affecting the other. Eq. (2.42) and (2.43) use the measured drain current, I_D to derive, $(w_n,w_p)_{opt}$ and $(l_n,l_p)_{opt}$ such that the drain current is much closer to the designed bias current, J, without affecting gate capacitances. Similarly Eq. (2.44) and (2.45) use the measured gate capacitances, C_{gsp} and C_{gsn} to make the second iterations closer to the designed gate capacitance, C_{gs}. Initial and improved dimensions can be found in Table 2.7.

$$(w_n, w_p)_{opt} = \sqrt{\frac{J}{I_D}} \cdot (w_n, w_p)_{init} \qquad (2.42)$$

$$(l_n, l_p)_{opt} = \sqrt{\frac{I_D}{J}} \cdot (l_n, l_p)_{init} \qquad (2.43)$$

$$(w_n, l_n)_{opt} = \sqrt{\frac{C_{gs}}{C_{gsn}}} \cdot (w_n, l_n)_{init} \qquad (2.44)$$

$$(w_p, l_p)_{opt} = \sqrt{\frac{C_{gs}}{C_{gsp}}} \cdot (w_p, l_p)_{init} \qquad (2.45)$$

The settling response of the delay cell will be overdamped for negative inputs and underdamped for positive inputs [87] and a convenient way to improve on the initial memory switch dimensions is to use simulation to plot the settling of the memory transistor drain currents. By varying the memory switch width and repeating this

simulation, the user can choose the curve and hence width which gives the best settling response.

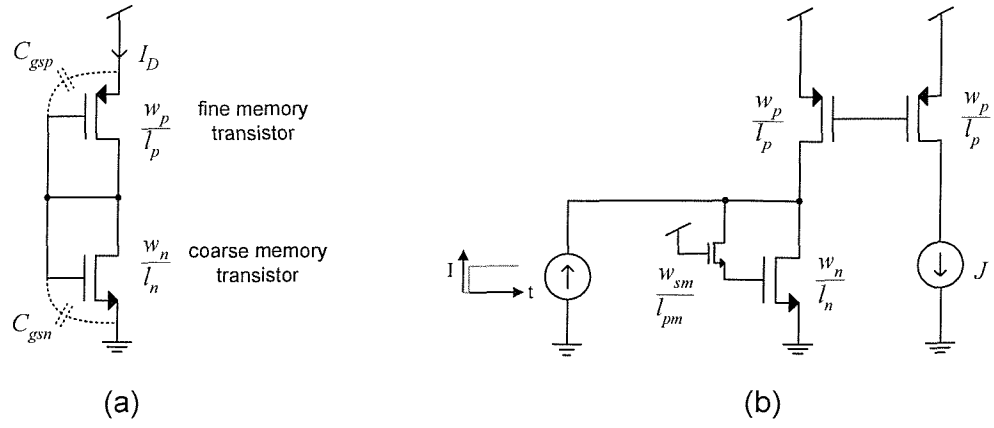


Figure 2.16 Test circuits for optimising S^2I operating point (a) and settling (b)

2.4.2 Adaptor blocks

Adaptors are fundamental building blocks in wave filters (Table 2.2) and therefore careful design is necessary to obtain good filter performance. Adaptors consist entirely of current mirrors and in theory this should not present a major design challenge. However, the required current mirrors in the adaptors need to simultaneously meet a number of conflicting design criteria including high output resistance to reduce conductance ratio errors, and a high output voltage swing to minimise distortion. Whilst a current mirror with high output impedance can be obtained using a simple cascode topology, this limits the output voltage swing since two more threshold voltages are lost in the voltage headroom [102]. In addition to the two above requirements, the bandwidth of the current mirror must be considered as well as the area overhead used by the design. The design of the current mirrors needed for adaptor blocks is considered next.

The use of high compliance cascode current mirrors (Figure 2.17(a)) increases output resistance and allows a minimum output voltage of just $2V_{gt}$, which is the smallest possible for a cascode design [102]. The output resistance of the high compliance current mirror is given by Eq. (2.46), where r_{out} is the output resistance of the current mirror, g_{m2} is the small signal transconductance of the cascode transistor and

r_{ds1} and r_{ds2} are the small signal drain source resistances of the mirror and cascode transistor respectively. The transconductance of an NMOS transistor is given by Eq. (2.47), where β_n is a process parameter, I_D is the large signal bias current and W and L are the transistor dimensions. The small signal transistor channel resistance is given by Eq. (2.48), where λ is the channel length modulation parameter, which is inversely proportional to the transistor length, L.

$$r_{out} \cong g_{m2} r_{ds2} r_{ds1} \tag{2.46}$$

$$g_m = \sqrt{2\beta_n \frac{W}{L} I_D} \tag{2.47}$$

$$r_{ds} = \frac{1}{\lambda I_D} \tag{2.48}$$

The high compliance design is such that the minimum output voltage will be equal to $2V_{gt}$, and so to increase the maximum swing it is necessary to decrease this value:

$$V_{gt} = \sqrt{\frac{2I_D L}{\beta_n W}} \tag{2.49}$$

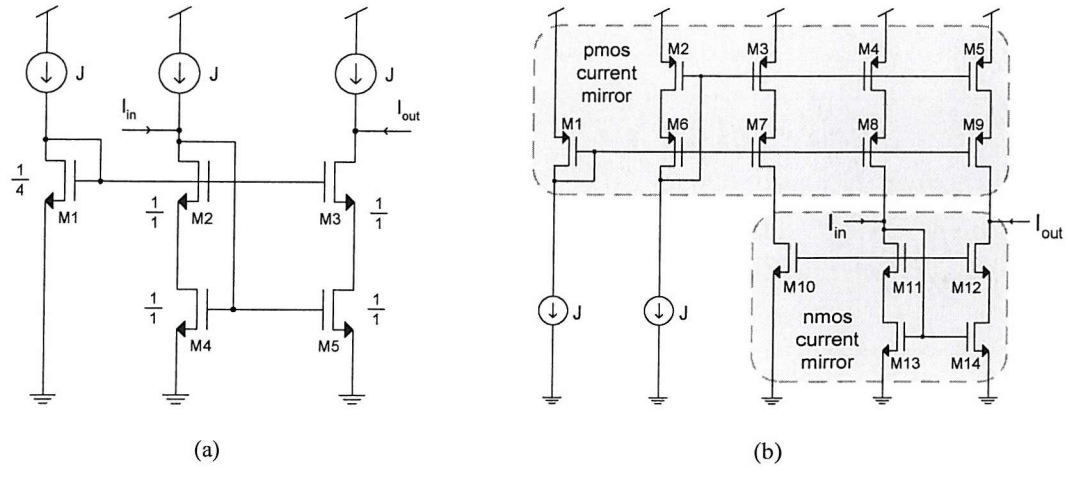


Figure 2.17 High compliance current mirror (a), with PMOS current sources (b)

It has been found through numerous wave filter designs of different requirements that, having taken into account adaptor coefficients and scaling, current mirror ratios in wave filters are unlikely to be more than 1:3. For a simple one output unity gain current mirror, the mirror bandwidth is determined by the diode connected memory transistor g_m and the capacitance at the input node as shown in Eq. (2.50) [103]. The gate source capacitances of the current mirror transistors, C_{gs1} and C_{gs2} are given by Eq. (2.51), which leads to Eq. (2.52).

$$BW_{(-3dB)} \approx \frac{g_{m1}}{C_{gs1} + C_{gs2}} \quad (2.50)$$

$$C_{gs} = C_{ox} WL \quad (2.51)$$

$$BW_{(-3dB)} \approx \frac{\sqrt{2\beta_n I_D}}{2L\sqrt{L}\sqrt{W}C_{ox}} \quad (2.52)$$

A bandwidth of 100MHz is chosen, which is 100 times the sampling frequency to ensure fast cell settling and correct operation. The current mirrors can have up to three output branches and ratios of up to 1:3 and with this in mind, the total capacitance at the diode connected node could be ten times the basic C_{gs}, and as such Eq. (2.52) is used with 10 instead of 2 in its denominator. With a β_n of 77μA/V^2, I_D of 200μA and C_{ox} of 2.1fF/μm2 and using an estimated width of 100μm gives a length of 5.189μm. From this the mirror transistor lengths are chosen to be 5μm (rounded dimensions will later improve ease of layout). In practice the length of the cascodes is made smaller, so as to reduce their V_{gt}, and a compromise exists between this and reducing their output resistance. For this case study, a value of about a third of the mirror transistor length is chosen, giving 1.5μm.

The load resistance of the current mirror will dictate what the voltage swing on the output will be over the signal range, and in this case study this load is likely to be that of the input to another current mirror, in the region of a few kΩ. With a 200μA

maximum signal the output node voltage will not change much more than a few hundred mV. With this in mind, an output swing of 1V is chosen (0.5V in either direction), and assuming that the mirror will be designed to have a quiescent output level at half V_{dd}, this gives 1.15V ($\frac{1}{2}V_{dd}$-0.5V) for the gate overdrive voltages of the NMOS mirror transistor and its cascode. The chosen lengths of the cascode and mirror transistors are such that the V_{gt} for the cascodes can be about half that of the mirror transistors ($\sqrt{1.5/5}$). This means that the mirror transistor should be designed for a V_{gt} of about 0.77V and the cascode for 0.38V and now using Eq. (2.49), gives a width of 43.8μm for the mirror transistor. The layout will be greatly simplified if the same width is chosen for the cascodes, and since this will only incur a slight decrease in the output voltage swing it is acceptable. Initial dimensions for the mirror are therefore 43.8/5 for the NMOS mirror transistor and 43.8/1.5 for its cascodes. In practice the ideal current sources of Figure 2.17(a) are replaced with a PMOS current mirror, as shown in Figure 2.17(b) and a similar method can be used to obtain the initial dimensions for this mirror. The initial design has been optimised for ease of layout, where a common width factor can greatly assist in the transistor stack design discussed later in Section 2.4.5. These final dimensions are shown in Table 2.8.

Table 2.8 High compliance current mirror dimensions

Transistor	Width (μm)	Length (μm)
M2-5	120	5
M6-9	120	1.5
M11-12	40	1.5
M13-14	40	5
M10	10	5
M1	30	5

The input and output resistances for the current mirror are found to be 1.9kΩ and 8.7MΩ respectively, which gives a respectable conductance ratio of over 4500.

2.4.3 Bias generation

The high compliance current mirrors (Figure 2.17) and the S^2I memory cell (Figure 2.14) require bias voltage and clock generation circuitry for their correct operation,

which are detailed in this section for completeness. Figure 2.18 shows the biasing circuitry, with transistor dimensions given in Table 2.9. Transistor M1 mirrors the reference bias current to transistors M2 and M3, the latter of which acts as a current source for M8, setting up the correct NMOS cascode bias voltage, 'NCBIAS'. This is then used to set up the correct gate voltages for the NMOS mirror formed by M9 to M16 which acts as three current sources with reference current mirrored through M3. The first current source is used to generate the PMOS cascode bias voltage, 'PCBIAS' using the diode connected M6, and the second current source is used to generate the PMOS mirror bias voltage 'PBIAS' through the M4 and M7. Finally, the last current source is used with M5 to generate the bias voltage for the memory cells, 'S2IBIAS'.

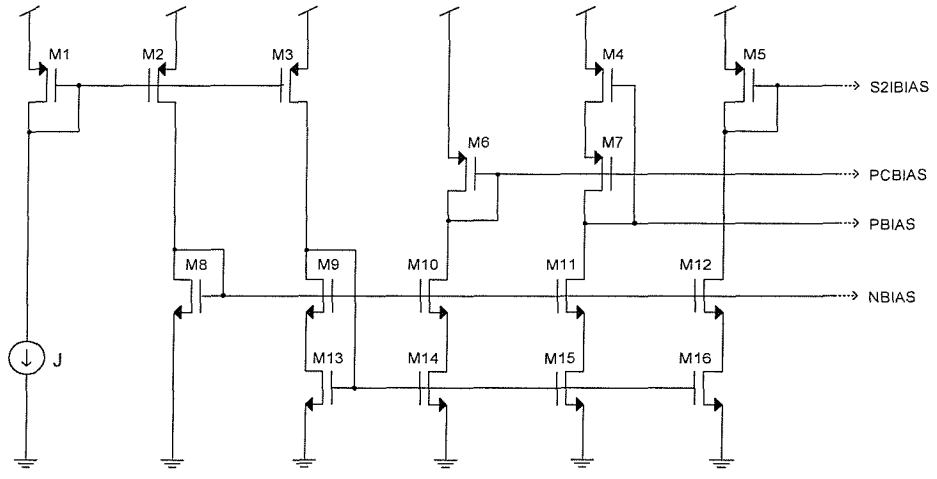


Figure 2.18 Biasing circuitry for entire filter

Table 2.9 Bias block transistor dimensions

Transistor	Width (μm)	Length (μm)
M1, M2, M3	120	5
M4	120	5
M5	150.3	20.95
M6	30	5
M7	120	1.5
M8	10	5
M9-12	40	1.5
M13-16	40	5

2.4.4 Clock generation

The S^2I memory cell used in the case study (see Figure 2.14) is clocked by four phases, $\phi 1$, $\phi 1a$, $\phi 1b$ and $\phi 2$. A SI delay cell consists of two of these memory cells and as such requires six phases, $\phi 1$, $\phi 1a$, $\phi 1b$, $\phi 2$, $\phi 2a$ and $\phi 2b$. Special requirements are laid on these phases to ensure correct operation of the S^2I cell, as indicated on Figure 2.14. To design a circuit to achieve the exact timing constraints given by the S^2I delay cell requires a simple four phase ring counter, which can be designed using Karnaugh maps using four state variables, $Q0$-$Q3$. The next state logic for $Q0_n$-$Q3_n$ is found to be:

$$Q0_n = \overline{Q2}.\overline{Q1}.\overline{Q0} \qquad Q1_n = Q0 \qquad Q2_n = Q1.\overline{Q0} \qquad Q3_n = Q2.\overline{Q1}.\overline{Q0} \tag{2.53}$$

Standard gates are then used to logically implement the different S^2I phases, for example $\phi 1$ is the logical AND of the first two ripple counter outputs, and the subphases are the respective ripple stage outputs AND NOT the previous output, thus maintaining a non-overlapping nature:

$$\begin{aligned} &\phi 1 = Q0 + Q1 \qquad \phi 2 = Q2 + Q3 \qquad \phi 1a = Q0.\overline{Q3} \\ &\phi 1b = Q1.\overline{Q0} \qquad \phi 2a = Q2.\overline{Q1} \qquad \phi 2b = Q3.\overline{Q2} \end{aligned} \tag{2.54}$$

This approach to clock generation requires a clock of four times the sampling frequency, which for this case study would be 4MHz. Figure 2.19 shows the completed clock generator circuit, made purely from standard digital cells, with Figure 2.20 showing the correct phases at the case study sampling frequency of 1MHz (clock frequency of 4MHz). Figure 2.21 indicates that the exact timing relations required are achieved, which can be seen at a high sampling frequency of 125MHz. It is a good idea to check the timing relations are still maintained after a parasitic extraction of the designed chip, since at this stage it is difficult to estimate the capacitive loading on the clock lines. If the restraints no longer hold then dummy loading can be added to the clock lines to restore the exact timing relations.

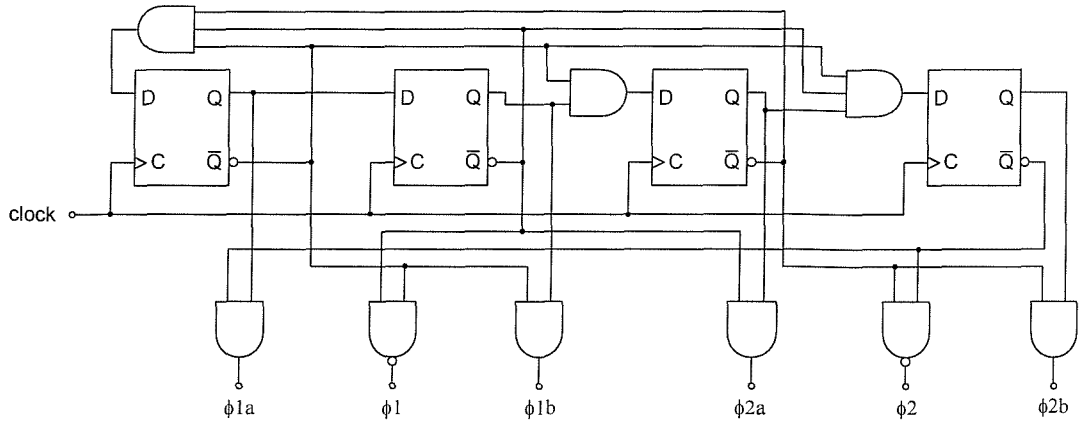


Figure 2.19 Six phase clock generation circuit

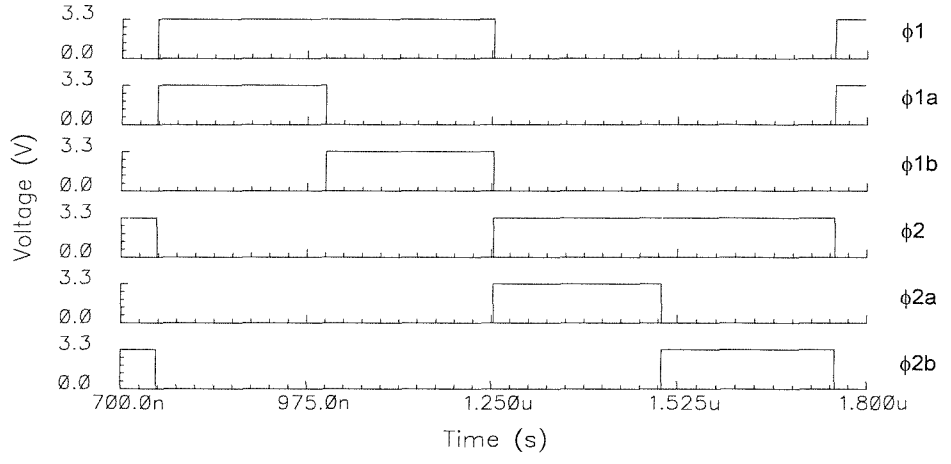


Figure 2.20 Clock phases and timing relations

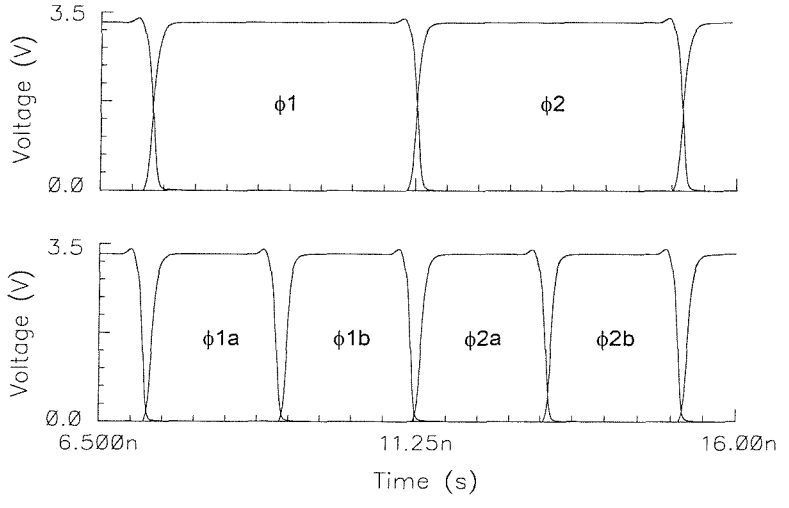


Figure 2.21 Exact timing relations of clock generation circuit

2.4.5 Layout strategy

In a full custom analogue layout, it is well known that absolute matching of different sized transistors in different locations is particularly poor, yet relative matching of identical and adjacent structures can be orders of magnitude better. Matching considerations must be taken into account if transistors in an analogue layout should be matched to better than a few percent [104]. Given that the entire filter structure consists of current mirrors and their ratios, matching is of utmost importance and it is therefore essential to consider a layout strategy to address this problem. Standard practice for improved matching involves splitting devices into a number of parallel transistors of a unit width, called gate strips, chosen such that all the different widths can be implemented by integer numbers of this unit. For example, three transistors of dimensions 50/2, 100/2 and 125/2 should be split into gate strips of 25, which would require 2, 4 and 5 identical unit gate strips respectively. Furthermore, common centroid layout techniques can completely cancel the effect of linear process parameter gradients and involves laying out devices with symmetry about the central axis, for example ABBA or ABAABA. A so-called stack of transistors can be made much more compact if adjacent drain/source diffusion regions of devices can be shared, which is the case when they are electrically connected. With this in mind, an Euler path is talked about in [105] whereby a continuous path is found from either *gnd* (for NMOS) or V_{dd} (for PMOS) through all the gate strips once and back to *gnd* or V_{dd}. This path then defines how the gate strips will be laid next to each other allowing all adjacent drain/sources to share common diffusion regions. The stack of transistors is electrically connected as a current mirror at a later stage.

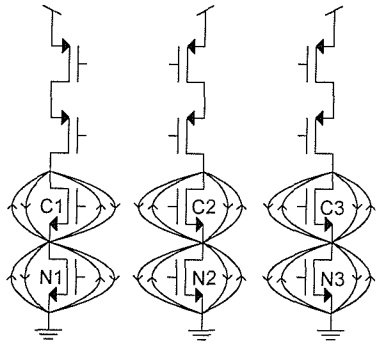


Figure 2.22 Current mirror Euler path

For example, consider a two output current mirror of the kind used to implement the adaptor blocks (Table 2.6), and shown in Figure 2.22 as just the transistor stacks. The NMOS memory transistors, N1 to N3 and their cascodes C1 to C3 should all be well matched to ensure good current mirror performance. For just the input current mirror branch, C1 and N1, a simple Euler path stack can be generated, by splitting each transistor into two and abutting as follows:

N1 C1 C1 N1

This is indeed common centroid, but when first output branch is added, the stack becomes:

N1 C1 C1 N1 N2 C2 C2 N2

Which is clearly not common centroid since it is not symmetric about the centre. This is why the transistors have been split into four gate strips, as indicated by the drawn arrows on Figure 2.22, and the resulting arrangement for the entire two output branch current mirror becomes:

D N3 C3 C3 N3 N2 C2 C2 N2 N1 C1 C1 N1 N1 C1 C1 N1 N2 C2 C2 N2 N3 C3 C3 N3 D

This stack is true common centroid and as such the unit gate width for N1 to N3 and C1 to C3 is a quarter of their full width. A dummy gate strip, D, is placed either end of the stack to prevent unmatched undercut on the edge of the gate poly of the most outer N3 gate strips. The gate, source and drain of these dummy transistors should all be connected to *gnd* (or V_{dd} for the PMOS stack) and their length can be the minimum allowable by the technology. In the final layout all transistors are made from unit gate strips, which are interdigitated, and in a common centroid arrangement with dummy strips at the ends. Furthermore, all gate, source and drain regions have the same extension, and guard rings around both the NMOS and PMOS stack ensure bulk potential. Figure 2.24 shows the three layout stages for a simple one output current mirror:

Stage 1: determine unit transistor length and stack in terms of gate strip arrangement.

Stage 2: overlap suitable NMOS unit layout cells such that left half of this stack is built, add a suitable dummy cell to the left end of this half stack and spot on connections for the NMOS drain gate and source connections. Repeat this for the left half of PMOS stack and place above the NMOS half stack.

Stage 3: select NMOS and PMOS half stacks and mirror this to complete full stack. Add NMOS and PMOS guard rings and external routing as appropriate, leading to a complete mirror layout.

Delay cells are laid out in two separate parts, a transistor stack, using the method just described and the delay cell switches. The switches are in a physically separate area, with their own guard ring to prevent coupling of the clock signals through the substrate. Careful arrangement and abutting of the switches can lead to an efficient layout, which can be placed between the delay cell transistor stack and the clock generating circuitry. This layout is depicted in Figure 2.23 (fixed switch sizes have been illustrated for clarity). The area between the group of switches and the delay cell transistor stack is such that a wide substrate tap can be placed at a later stage if necessary, to further prevent digital clock signals coupling though the substrate. Further details concerning the layout of all parts of the prototype chip can be found in Appendix A.

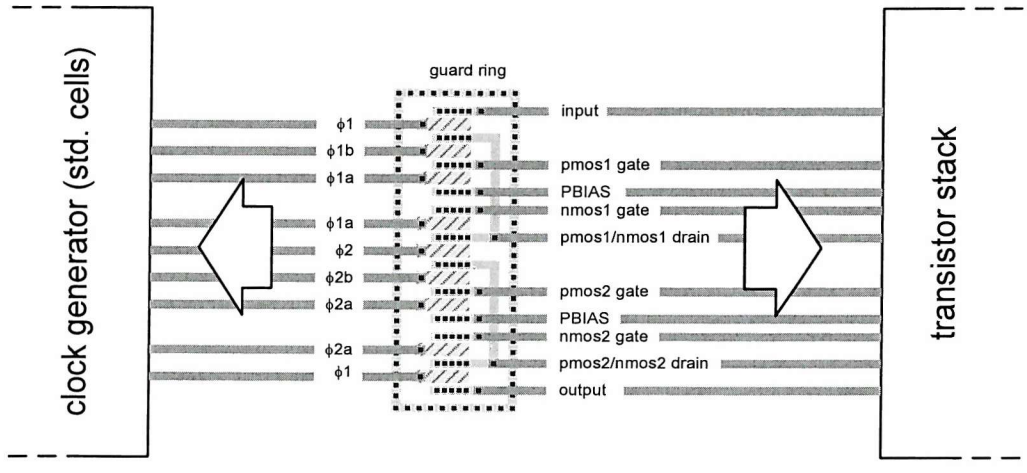


Figure 2.23 Delay cell layout

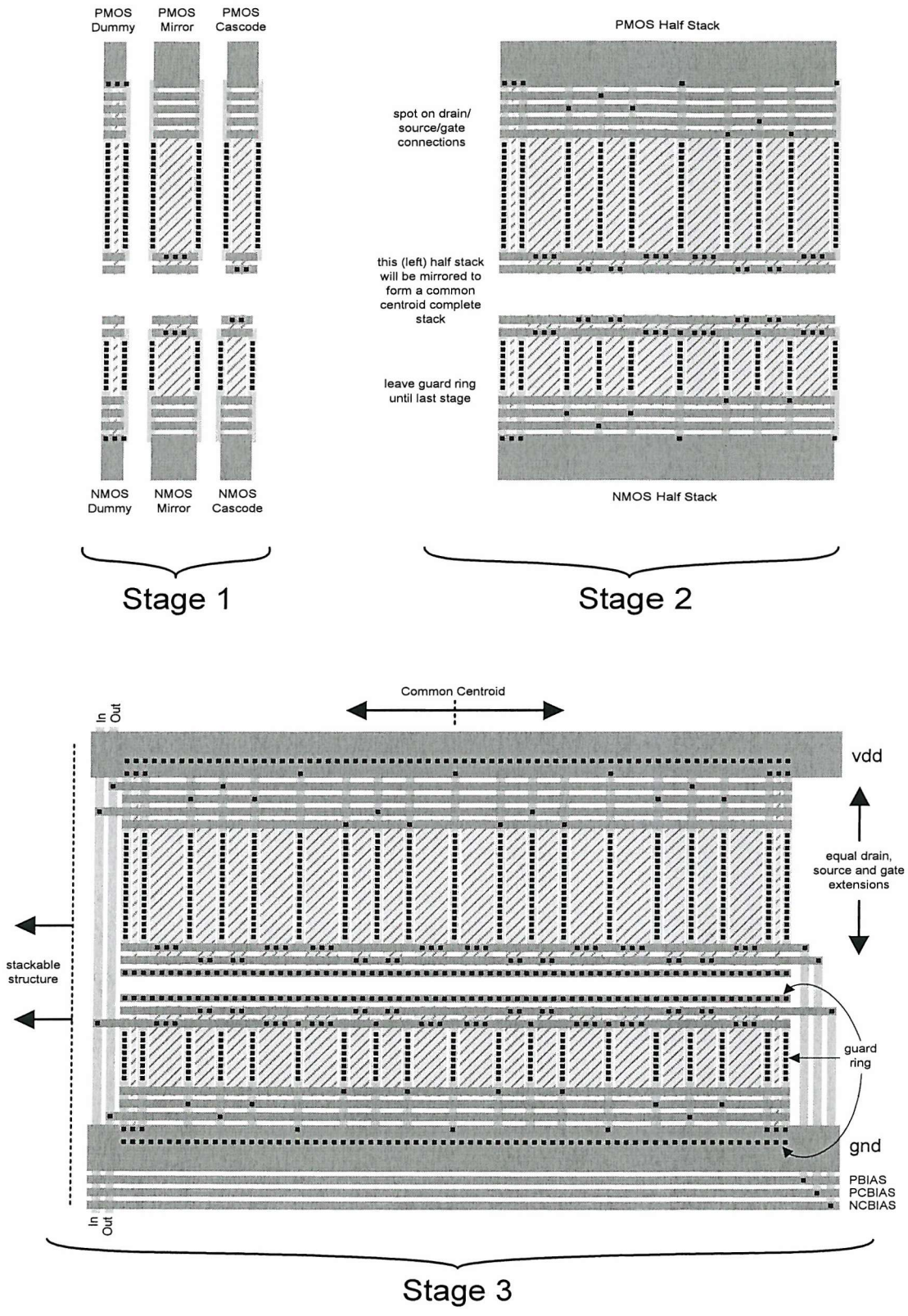


Figure 2.24 Example current mirror layout

2.5 Experimental Results

Detailed results of the case study developed throughout this chapter are given in Section 2.5.1. Further case studies used to verify the success of the power aware scaling approach are given in Section 2.5.2. Extended results for the prototype chip can be found in Appendix B.

2.5.1 Prototype filter chip

To validate the design flow and techniques proposed in this chapter, a 3rd order lowpass elliptic filter was designed and fabricated, to specifications detailed in Section 2.1.3 and shown in Table 2.10. The frequency response of the transistor level design of the filter, simulated using BSim3v3 foundry models is shown in Figure 2.25 together with the ideal frequency response.

Table 2.10 Designed and measured performances for the filter case study

Specification:	Designed	Measured:
Filter type:	3rd order lowpass elliptic	3rd order lowpass elliptic
Passband ripple:	0.5dB	0.8dB
Stopband attenuation:	≥ 30dB	29dB
-3dB frequency:	100kHz	98kHz
Sampling frequency:	1MHz	1MHz
SNR:	≥ 60dB	64dB
Supply voltage:	3.3V	3.3V
Nominal bias current:	200μA	200μA
Power:	≤ 50mW	45mW
Technology:	0.6μm single poly CMOS	0.6μm single poly CMOS

As can be seen there is an excellent correlation between the ideal and transistor level results. This can be attributed to the optimisation of the wave adaptor coefficients (Section 2.2.3) and modifications to improve on the initial transistor dimensions, which were derived from square-law saturated models. As outlined in Section 2.3 the spread of internal signal levels can produce distortion and the adaptor current mirror scaling was essential to address this problem in the final filter realisation. This is

evidenced by the presence of a 6% error in the filter cut-off frequency and 1dB insertion loss when the filter was simulated using the initial wave adaptor coefficients, delay cell dimensions and no current mirror scaling.

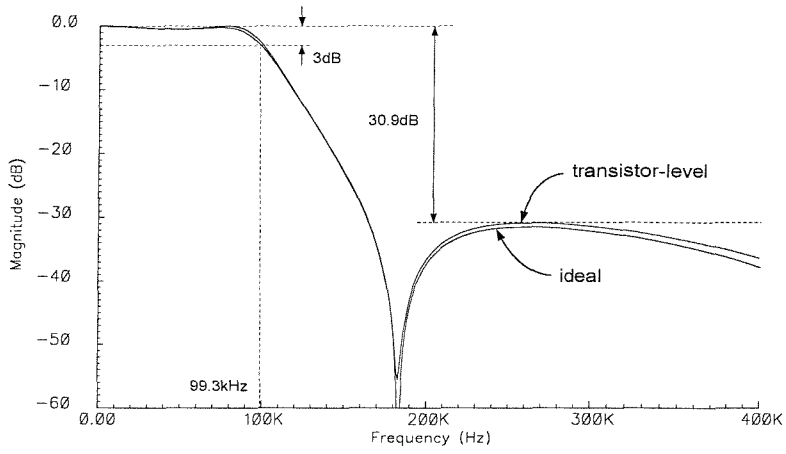


Figure 2.25 Simulated response of transistor level filter

The prototype filter chip has a current mode transfer function (I_{out}/I_{in}) and to facilitate measurements using voltage domain instruments, a 50kΩ resistor was used at the input of the filter as a simple V/I converter and a 5kΩ resistor was used at the output buffered by an op amp as a I/V converter. The bias voltages, PCBIAS, PBIAS, NCBIAS and S2BIAS, needed for the delay cells and current mirrors are generated from a single off-chip current reference (Section 2.4.3) and were measured as 0.867V, 1.70V, 2.47V and 1.67V respectively. Note the filter has a sampling-to-cut-off frequency ratio of 10:1, but the clock frequency is four times the sampling frequency as required by the clock generator for the S^2I delay cell (Section 2.4.4).

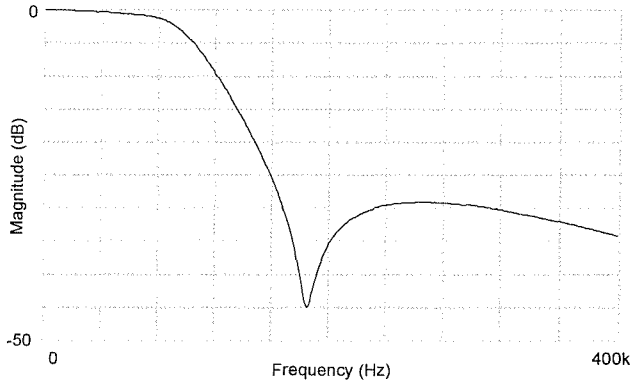


Figure 2.26 Measured chip response with 1MHz sampling frequency

The measured response of the prototype filter chip is shown in Figure 2.26. The filter frequency response is close to the simulated, with a mean cut-off frequency of 98.2kHz compared to 100kHz (ideal), and up to 0.8dB pass-band ripple compared to 0.5dB (ideal). From twenty samples, the cut-off frequency spread was found to be ±0.66% from the mean of -1.79%. The discrepancies between simulated transistor level design and measured results can be attributed to accuracy problems in realizing non-integer current mirrors ratios and normal tolerances associated with the fabrication process. It is interesting to note that the measured filter has a stopband notch at 185kHz compared to the simulated notch at 183kHz. This close agreement together with the well-defined notch of -45dB depth and stopband attenuation of -29dB is attributed to the careful layout strategy (Section 2.4.5).

A useful feature of SI filters is the ability to vary their bandwidth with an external clock, similar to switched capacitor (SC) filters. Figure 2.27 shows measured filter bandwidths from 10kHz to 100kHz obtained with different clock frequencies. Again, there is good agreement between the filter shape of the simulated and measured frequency responses. Figure 2.28 shows the filter response with a sampling frequency of 2.5MHz, which is 2.5 times greater than the maximum designed sampling frequency. The corresponding filter cut-off frequency is 243kHz (compared to 250kHZ ideal) and despite an expected reduction in stopband attenuation (-27dB) and slightly increased passband ripple (1.8dB), the filter shape is still reasonable.

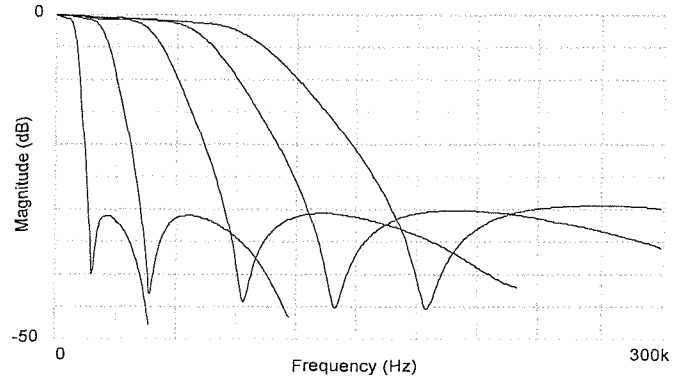


Figure 2.27 Measured chip response with different sampling frequencies

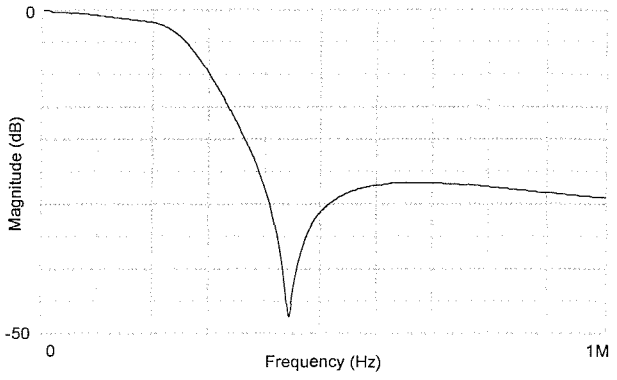


Figure 2.28 Measured chip response with 2.5MHz sampling frequency

Sampled-data filters such as SI and SC are known to have inferior noise performance when compared with continuous-time filters. To give insight into the noise performance of the SI filter, Figure 2.29 shows the simulated PSD and integral noise, computed as a transfer function from all the noise sources in the circuit to the output node. The integral noise levels off at 70nA which gives a signal to noise ratio (SNR) of 66.62dB. Figure 2.30 shows the measured noise of the filter at the output, with the power on and off, and this compares well with the simulated results.

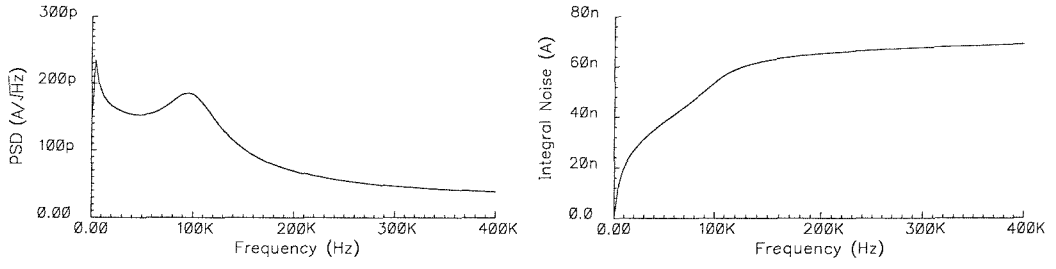


Figure 2.29 Simulated noise of transistor level filter

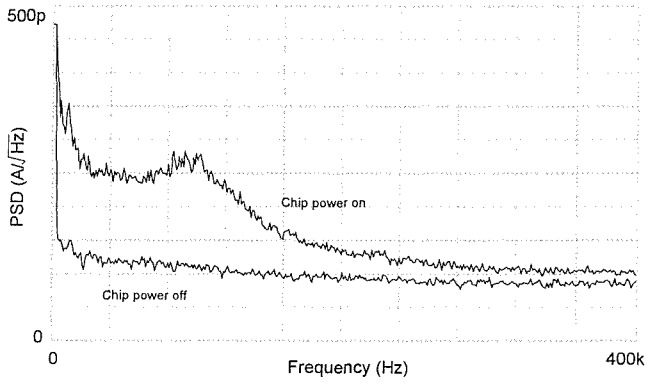


Figure 2.30 Measured noise of prototype filter chip

The measured SNR calculated from the measured noise response is 64dB which represents good performance from a sampled-data filter and is mainly due to optimised memory cell design, careful scaling of bias currents (Section 2.3) and suitable layout considerations.

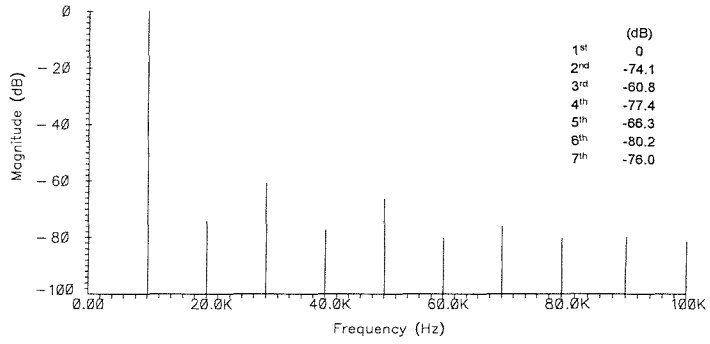


Figure 2.31 Simulated distortion of transistor level filter

Figure 2.31 shows the simulated distortion of the transistor level filter with a 1kHz input of 50μA amplitude. Figure 2.32 shows the measured distortion for the same input signal where it can be seen that the harmonic content of the output signal appears greater than the simulated response. This is likely to be due to the test setup, and in particular the use of a simple resistor at the input and output of the filter for V/I conversion.

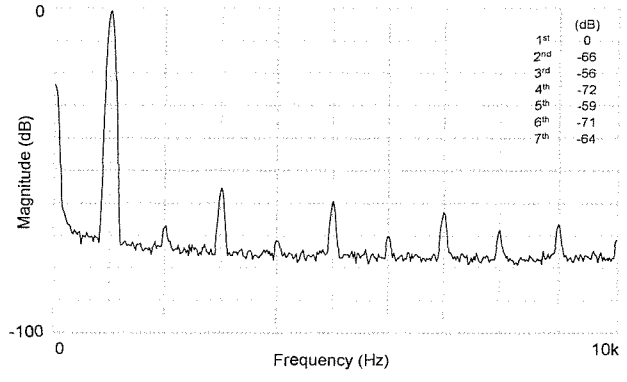


Figure 2.32 Measured distortion of prototype filter chip

The measured power consumption of the filter, bias and clock structures is 45mW which indicates how scaling (Section 2.3) has reduced the estimated power. A chip photo is shown in Figure 2.33 and the filter core size is $0.75\mu m^2$.

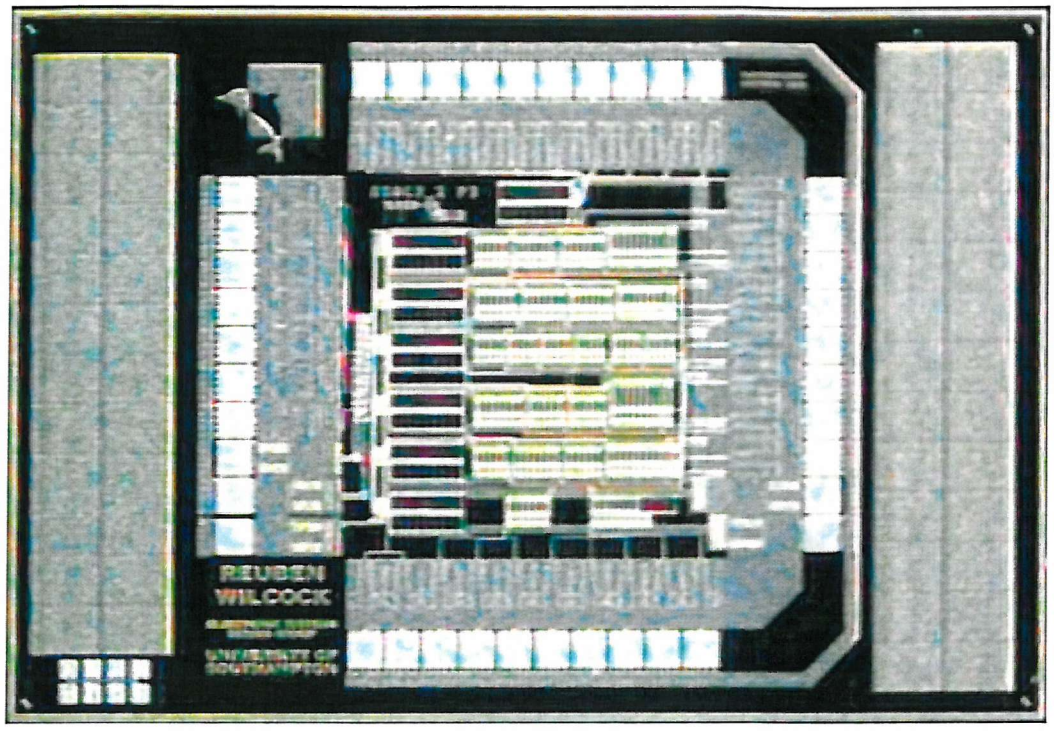


Figure 2.33 Die photograph

The measured performances of the test chip are summarised in Table 2.10. To put in context the results of Table 2.10 with previous work in SI wave filters, the presented low pass filter has the highest bandwidth reported to date for fabricated SI filters using the wave synthesis technique, which has been possible due to high performance adaptor block and delay cell design in Section 2.4 and highly matched layout. The power consumption of 45mW is also considerably lower than the 160mW reported for the integrator based SI filter of [13], which can be attributed to careful choice of bias current (Section 2.4) and bias current scaling (Section 2.3).

2.5.2 Power aware results

To verify the success of the power-aware scaling approach outlined in Section 2.3, two case studies were taken to real transistor level, using 0.6μm 3.3V BSim3v3 CMOS foundry models. The filters chosen were a 3rd order elliptic and 5th order Chebyshev lowpass filter of wave structures given in Figure 2.34(a) and Figure 2.34(b) respectively. The transistor level structure for the elliptic filter case study has already been shown in Figure 2.13 (Page 46).

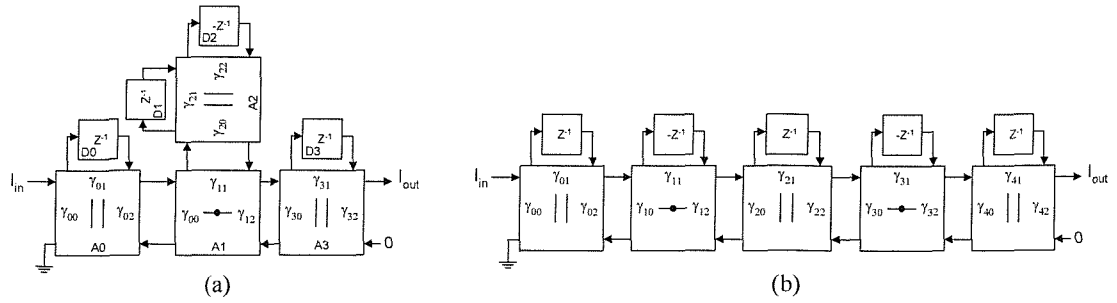


Figure 2.34 Wave structures for elliptic (a) and Chebyshev (b) case study

In order to quantify the extent to which internal blocks are distorting, a spectrum of the filter output signal for a given input signal can be used to calculate a THD figure. Clearly, distortion increases as signal levels increase and transistors begin to leave the saturation region. In the designed filters the worst case normal working distortion would therefore arise when the input signal is at its maximum value, i_{max}. Therefore, calculating the THD with the maximum input signal of i_{max} gives us a good metric to compare the extent to which internal blocks are exceeding their dynamic range.

Table 2.11 Worst case distortion with no scaling and maximum input signal

Elliptic:		Chebyshev:	
2nd harmonic	-50.8dB	2nd harmonic	-27.4dB
3rd harmonic	-24.3dB	3rd harmonic	-27.1dB
4th harmonic	-55.4dB	4th harmonic	-40.1dB
5th harmonic	-43.9dB	5th harmonic	-51.4dB
THD:	6.13%	THD:	6.21%

The wave filter structures were derived from their passive reference prototypes and adaptor coefficients calculated [12]. Initial simulations were run having not scaled any parts of the filter and with a signal input equal to i_{max} so as to demonstrate the resulting worst case distortion. Table 2.11 shows the first five harmonic levels for the two filters, and the total harmonic distortion calculated from the first seven (harmonics higher than this are insignificant). Despite this being a worst case measurement, the high THD levels of around 6% clearly indicate that without any scaling some of the internal nodes are exceeding their dynamic range with the maximum designed input, i_{max}. To illustrate the non-homogeneous signal levels in the filters, internal node responses at points *X*, *Y* and *Z* of the transistor level elliptic filter

of Figure 2.13 (Page 46) are shown in Figure 2.35. The response at points *X* and *Y* show that the bias value is being exceeded at some frequencies, and response *Y* in particular demonstrates that some current mirrors would be running at very distorted levels for the entirety of the passband. The response at point *Z* demonstrates how some bias values exceed the signal values they support, as is discussed in 2.3.

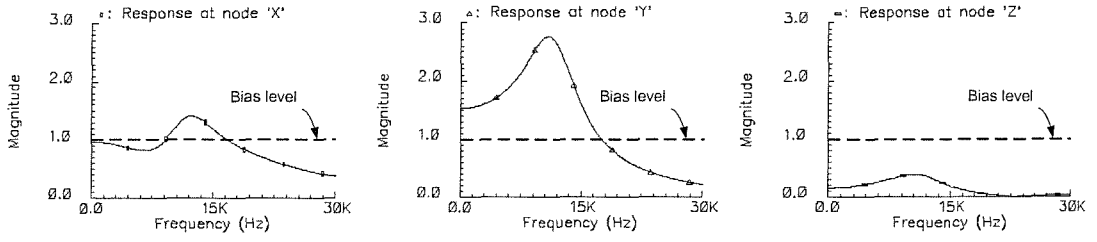


Figure 2.35 Internal signal levels in the wave filter example

All internal signal levels for the filters were then determined as described in Section 2.3 and the signal and bias levels in the filter scaled *only* where distortion occurs, as in [11]. A quiescent power reading was taken and then the full proposed method was used, which should not only prevent block level distortion but also save power by reducing bias currents where the signals they support are small. A second quiescent power reading was then taken and the saving in power calculated. Shown in Table 2.12 are the results from these simulations. It is clear that using scaling has greatly reduced the distortion in both cases, showing that this is therefore an essential part of the wave filter design process. However, the results also clearly show that by using the proposed method it is possible to not only prevent block level distortion, to at least the same extent as simple scaling, but also to save power as part of this process.

Table 2.12 Power savings in both case studies

	Elliptic:		Chebyshev:	
Method:	Basic Scaling [11]	Proposed method	Basic Scaling [11]	Proposed method
THD:	2.5%	1.6%	1.38%	1.32%
Power:	24.5mW	20.4mW	48.8mW	45.8mW
	Power saving:	16.6%	Power saving:	6.1%

For the two case studies, the power savings are 16.6% for the 3rd order elliptic filter and 6.1% for the 5th order Chebyshev filter. In order to validate the noise

considerations of Section 2.3.3 a periodic noise analysis, using SpectreRF, was run for both case studies and the integral noise spectrums plotted. As can be seen in Figure 2.36, noise is reduced in line with the general reduction in power, with the total reductions in RMS noise being 14.7% and 9.7% for the elliptic and Chebyshev case study respectively.

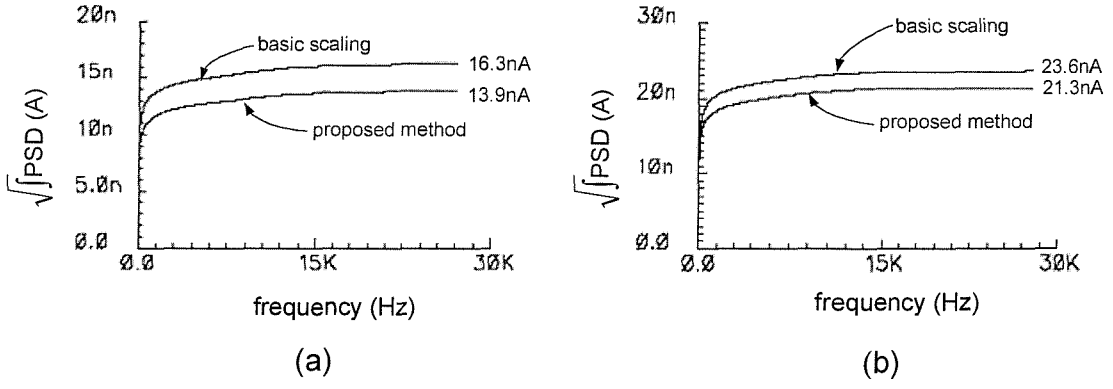


Figure 2.36 Noise performance for the elliptic (a) and Chebyshev (b) case study

The elliptic case study is similar to the fabricated prototype chip and by altering the bias current input, it has been possible to confirm the simulated results. The measured power for the filter core plus structures for test purposes is 22.2mW. The extra structures consume a total of 1.9mW (found through simulation), and hence the measured consumption of the filter alone is 20.3mW, which compares well with the simulated results of Table 2.12. A die photograph is shown in Figure 2.33.

To investigate the potential benefits of the method further, a larger range of filter orders were designed, and the power savings determined. The filters were first scaled only to prevent distortion, and then scaled using the proposed two stage method, including reducing bias currents where the signal levels are small. Table 2.13 shows the power savings for different filter types and orders when compared to using basic scaling and uniform bias currents as in [11, 12]. These savings can be as much as 16.6%, as verified through the elliptic filter case study. The figures show that power savings increase as the filter order decreases. Through extensive simulations it was observed that the greatest power savings occurred at the beginning and end of the wave filter structure, mostly in the first and last adaptor. As filter order increases, the

filter structure naturally becomes larger and so these savings become smaller relative to the overall filter power consumption. In any case as was shown in 2.3, the savings are in addition to the usual goal of preventing distortion.

Table 2.13 Power savings for further example filters

Filter type	Filter Order	Power saving (%)
Chebyshev	3	14.3%
Chebyshev	5	6.1%
Chebyshev	7	3.9%
Chebyshev	9	2.6%
Elliptic	3	16.6%
Elliptic	5	7.8%
Elliptic	7	4.8%
Elliptic	9	2.4%

2.6 Concluding remarks

This chapter has presented a design flow, from specification to layout, for SI wave analogue filters. Techniques that facilitate efficient mapping of wave adaptor coefficients to transistor level current-mirror designs together with an analytical approach to the transistor level design of the main wave filter building blocks including adaptors have been proposed. A two stage bias and signal scaling method has been presented which not only resolves block level distortion issues, thus improving filter THD but, unlike [11], addresses power consumption as part of this process. This power-aware method achieves significant power savings by taking into account maximum signal levels when assigning bias currents. The important issue of transistor matching in SI filters during the physical layout stage has been addressed through the development of a three-stage layout strategy. The proposed design flow has been demonstrated and validated by the design, fabrication and detailed characterisation of a 3rd order lowpass elliptic filter. The filter has the highest bandwidth reported to date for fabricated SI filters using the wave synthesis technique. Furthermore, it has been demonstrated that the proposed method achieves its goal of reduced power consumption by as much as 16.6%, with these savings verified using simulation and measurements taken from the fabricated prototype.

Chapter 3

AutoSIF: A CAD Methodology for Switched-Current Filters

Advances in technology have allowed complex systems consisting of many functional blocks to be integrated onto single system on chips (SoCs). One area of SoC realisation posing a considerable challenge is providing off the shelf analogue circuit functions in a similar manner to digital intellectual property (IP) cores [2]. In Chapter 1, Section 1.1, it was suggested that the switched-current (SI) technique was particularly suitable for SoC applications, tolerating the low supply voltages of modern processes and not requiring integrated passive components. However, despite the potential of SI, a considerable obstacle lying in the path of its acceptance is the difficulty and unfamiliarly associated with its design. In this chapter, it is suggested that a certain degree of design automation, including both tools and cell libraries may be the key to facilitate the recognition of SI. Section 3.1 gives a preliminary review of significant existing CAD tools and automation approaches in the literature, in order to identify their strengths and weaknesses. In Section 3.2 a generic CAD methodology is proposed comprising two parts, a hierarchical analogue cell library and a CAD tool. This methodology has been implemented for the SI wave filter design process given in Chapter 2 in the form of AutoSIF, a cell library

and tool which facilitates the rapid generation of analogue SI wave filters. The AutoSIF cell library is detailed in Section 3.3.1 and Sections 3.3.2 to 3.3.6 describe the accompanying CAD tool. Section 3.4 demonstrates the use of AutoSIF to generate an example filter design, and finally, Section 3.5 summarises the results and contributions of this chapter.

3.1 Preliminary review

Numerous CAD tools have been presented in the literature, for countless circuit types and functions. Many of these tools, especially those from academic research rather than industrial backgrounds, were developed from simple procedures used to aid the design of circuit parameters and gathering of results. A number of CAD tools have been presented in the literature specifically aimed at SI circuit design, including for filters [13, 84-86], data converters [97], and single memory cell building blocks [106]. These and other pertinent examples of existing CAD tools are now discussed in more detail so as to determine key factors contributing to the more successful methodologies.

As outlined in Chapter 2, Section 2.1.1, SI wave filters were first presented in the literature in the early 1990s in a number of publications by Yufera *et al* [10, 11, 29, 33, 34]. Although these authors did not explicitly discuss any CAD tools they developed, a brief mention of a tool named WAVER was made in [34] in 1994 which reportedly used high level wave models for coefficient calculation. No further details were published. Hughes *et al* were the first to apply the SI technique to sampled data filters in [7] in 1989 and following a large number of further publications [26, 32, 107, 108] an automated CAD tool for SI integrator based filter design was presented in 1996 [13]. This tool was a product of SCADS, a joint academic and industrial project led by Philips Semiconductors. As to be expected with a commercially biased project, specific implementation details for the tool were sparse, but it is likely from the information provided in [13] that the system was built into Cadence Design Framework II (DFII), using the SKILL® language. At the time of writing, Cadence is one of the main industry standard EDA suites for full custom analogue and mixed-signal circuit design. SKILL® is the proprietary language used

to implement Cadence and to a limited degree is available to designers for the development of programs which run within, and interface to, Cadence. The Phillips SCADS tool designed filters based on signal flow graphs (SFGs) which were then transposed to the sampled domain and implemented using SI bilinear integrators based on the S^2I memory cell proposed by Hughes *et al* in 1993 [26]. The SCADS project produced a number of video frequency filter examples [13].

At the heart of any SI filter design tool are procedures required to size a SI memory cell, and in 2000 Conner *et al* presented a tool named AZIMOV intended solely for this purpose [106]. AZIMOV supported the systematic design of SI memory cells and was capable of topology selection and transistor sizing from a set of user specifications. Analytical models, coupled with heuristic rule-based numerical optimisation were used with models for common memory cell structures. The tool is stand alone and was written in the C programming language which does not allow its integration into the major industry standard tool flows.

Another tool for the design of SI filters, named SYSCUF, was presented by Barau *et al* in 2001 [85]. The designed filters comprised of cascaded SI biquad sections made from second generation memory cells, and no simulated filter performances were given. SYSCUF was written in the C programming language. Also in 2001, Qingyun *et al* presented a tool named SIFDS for the design of SI filters [86]. SIFDS uses filter specifications to compute a transfer function and from this the corresponding SFG. Circuit level design was not covered but can be inferred to some degree from the SFG. The tool again appears to be implemented using the C programming language. Another SI filter design tool was presented by Handkiewicz *et al* in 2002 [84]. Three tools, Symb, Param and Layout dealt with three distinct aspects of the design process, the first two tools being used at the behavioural stage of the design and the third for layout generation of current mirrors. The designed filters were integrator based and no detail was given of the circuit level structures employed or overall filter responses. The tool was written in the C++ programming language.

A tool from industry was published by Xu *et al* in 2001, and although not intended for the design of SI filters is still relevant to this chapter [109]. This tool was similar to the SCADS project [97] in that it was integrated into Cadence using the SKILL®

scripting language. The tool was specifically aimed at creating analogue IP blocks and the entire principle was based around the use of 'pCells', parameterised layout cells in the Cadence layout editor. From basic pCells of a single device, larger sub-block pCells can be built, from which so-called 'ultra-pCells' are finally derived. The result is a design which is entirely described by a number of parameters, 73 in the example given, and these control the dimensions of every transistor in the layout. Defining the layout in a generic technology allows mapping of this to a specific technology when required.

A further tool not aimed at SI circuit design, but still relevant, was presented in 2002 by R. Castro-Lopez [110, 111]. A complete methodology was described for retargeting analogue blocks to different specifications or processes, again aimed at the IP core market. The tool made extensive use of pCells in Cadence and was written entirely in SKILL®. To design a cell, technology portable layout templates were first created, capable of being tuned with design constraints. Inputs to the tool included specifications and design goals which were used to derive the parameters in pCell definitions. The templates were manipulated and parameters passed down and inherited in a hierarchical scheme in order to generate the SKILL® code defining the designed cell. Retargeting from one technology to another was achieved by using a generic design rule database which could be mapped onto a range of specific technologies. Overheads involved in setting up the database for each new circuit are offset by the ease at which new specifications could be met.

From the wide variety of previous CAD tools and methodologies for SI filters and analogue IP cores, strong patterns contributing to the success of a number of the tools can be recognised. The next section discusses these considerations and proposes a CAD methodology based upon them.

3.2 Proposed CAD methodology

In this section a CAD methodology suitable for SI analogue IP cores is presented. Section 3.2.1 discusses the key features of the CAD tools presented in the literature and outlines the considerations of the proposed methodology based on these. The two

parts of the proposed methodology, a hierarchical analogue cell library, and a CAD tool are discussed in Section 3.2.2 and Section 3.2.3 respectively.

3.2.1 Key considerations

Examination of the previous work in Section 3.1 has led to the conclusion that a successful CAD methodology can be considered as two distinct parts, a carefully constructed library containing reusable, parameterised circuit cells, and a CAD tool, facilitating the design of these circuit cells. Furthermore, it seems that ultimately the usefulness of such a methodology may depend on the implementation details. Many tools presented in the literature were written as stand alone programs, in the C programming language [84-86, 106], which is a convenient choice given the fast development possible and numerous environments available. However, it is interesting to note that all the industrially related projects integrate the tool into major EDA tools, specifically Cadence [13, 97, 109-111], which is the most popular EDA tool for full custom front to back analogue and mixed-signal design. The provision of the LISP style proprietary language called SKILL® allows users to develop their own software which can be invoked from within Cadence. There are some clear advantages for integrating the proposed CAD methodology (cell library and tool) into Cadence:

- The tool would be written in SKILL® and so can be invoked seamlessly as part of a normal Cadence design flow.
- The cell library can be developed using Cadence.
- Results from the tool can be used to directly change design variables or assert schematic instance parameters in the cell library.
- The tool can access the cell library and can interactively invoke powerful proprietary simulation tools throughout the design flow.
- The methodology is far more likely to be used in practice, since a designer has it at his or her fingertips.

These benefits come at the cost of a significantly increased difficulty in implementing the tool, since SKILL® is a proprietary language and as such comes

with very little support, no programming environment and very little documentation. The following two sections discuss the generic cell library and tool in more detail.

3.2.2 Hierarchical analogue cell library

The key element of any successful circuit design tool is design reuse. The first part of the proposed methodology is a well-structured library, depicted in Figure 3.1, which contains a complete hierarchy of symbols and circuits to support the design of each major analogue core. A key decision in forming such a library is the number of levels of abstraction to provide, and in the proposed methodology most cores would have five levels, shown on Figure 3.1 and detailed follows:

Level 5: Top level symbol representing the entire core accompanied by associated high level parameters.

Level 4: High level functional blocks. High level parameters are resolved into mid level parameters, but not transistor dimensions at this stage.

Level 3: This level contains two implementations of the symbols in level 4, a behavioural model, plus a further low level block which hides the transistor level implementation of level 2. In the case where high level design parameters must be calculated and optimised it is best to stop at a behavioural circuit level, allowing fast simulations. When realistic simulations are required the transistor level would be used.

Level 2: Transistor circuit structure, allowing realistic simulations of the designed low level blocks.

Level 1: CMOS primitives and their associated BSim3v3 foundry models lie at the very bottom of the hierarchy.

Cadence allows powerful instance parameter passing through the pPar("") command which is used throughout the cell library. Transistor dimensions in levels 1 and 2 can be passed up to levels 3, 4 and 5 in an intelligent fashion, implementing design calculations in instance property fields. For example a property field of one of the

current mirrors in the adaptor block might contains 'pPar("Val1")*Val2+3', which passes a parameter 'Val1' from the higher level symbol to the instance, multiplies this by a global constant 'Val2' and adds a fixed value '3'.

Digital IP cores are readily traded in terms of process independent high level descriptions, which can later be synthesized into transistor level designs using standard libraries. It is far more difficult to produce analogue circuit designs in such a tradable format, for whilst digital circuits can gain performance from a shift to a smaller process, analogue circuits often lose performance or possibly stop working altogether [2]. Interestingly the use of SI facilitates technology portability in the cell library because only NMOS and PMOS transistors are required. Level 2 of the hierarchy is not committed to one particular transistor model, and it is a simple matter to exchange the MOS transistor in level 1 with one from a different process.

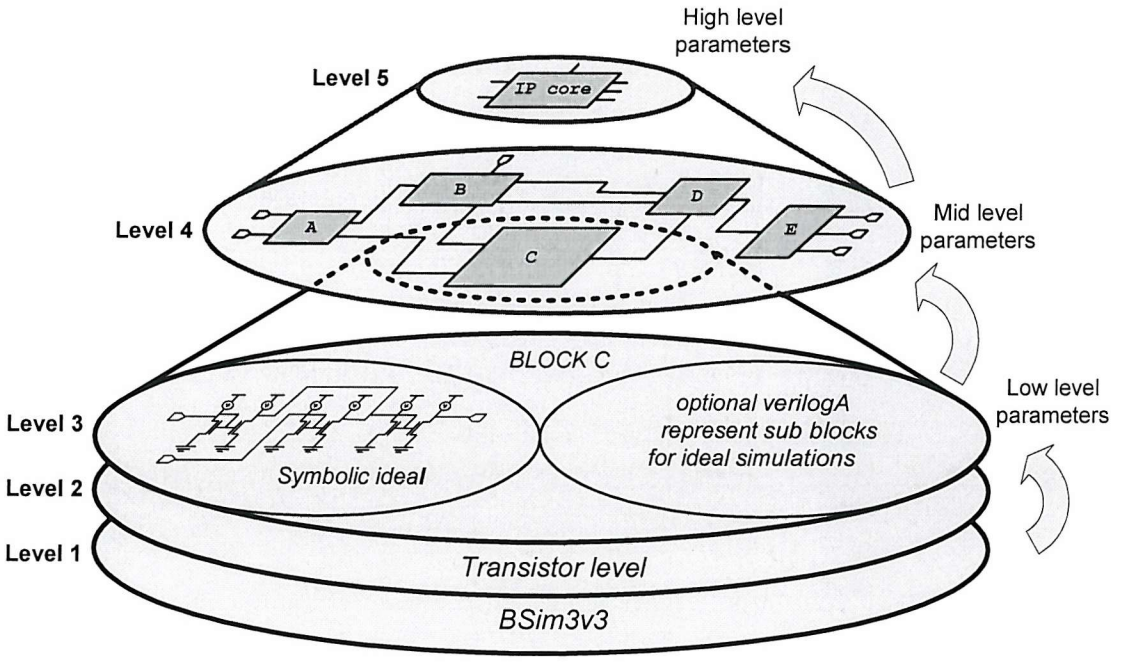


Figure 3.1 The hierarchical analogue cell library with parameter passing

3.2.3 SKILL® based CAD tool

The second part of the proposed methodology is a SKILL® based CAD tool to complement the analogue cell library, guiding the unfamiliar engineer through the design of analogue SI cores. The tool is driven by underlying equations relating

transistor parameters to specifications, rigorously capturing every design step and the main stages are shown in Figure 3.2 and detailed as follows:

Stage 1: Design of the system at block level using behavioural or ideal circuit descriptions. Ideal equations convert high level specifications into top-level design parameters, which can be optimised through algorithms if necessary.

Stage 2: Transistor level design of blocks in the ideal circuit, implementing the chosen high level design parameters. The main goal of this stage is to allow a user to explore the design space to achieve a first cut design, and subsequently optimise this based on transistor level simulations. It is at this middle step where most insight into SI design is gained since a user can easily examine tradeoffs without being hindered by numerous complex equations.

Stage 3: Having designed every sub block, the final stage is to put these together and run transistor level simulations for the entire design. Previous stages can be revisited if necessary and once the user is happy with the performance, the tool outputs a complete set of transistor dimensions and a fully defined schematic ready for layout.

Similar to the hierarchical cell library, where the use of SI has allowed it to be entirely technology independent with the exception of two symbolic transistor cells, the CAD tool facilitates technology portability through the use of a technology process file. This file contains a handful of process parameters required to achieve a first cut design and can be easily altered to one for an alternative technology.

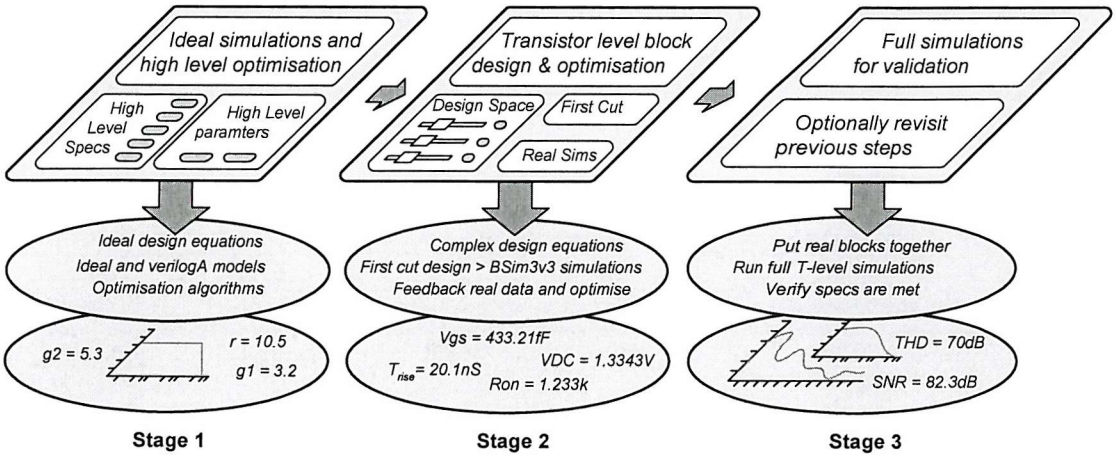


Figure 3.2 Proposed CAD tool steps

3.3 AutoSIF

The CAD methodology proposed in Section 3.2 has been implemented for the SI wave filters of Chapter 2 in the form of 'AutoSIF'. AutoSIF consists of a hierarchical analogue cell library, and a SKILL® based CAD tool. The cell library contains over 60 cellview schematics and the design tool consists of over 4000 lines of SKILL® code, numerous filter definition files, over 50 procedures and over 120 data structure components. AutoSIF automates the time consuming and systematic parts of the wave filter design flow detailed in Chapter 2, for example calculating wave coefficients, generating the initial memory cell design and performing the power aware scaling procedure. AutoSIF manages all the schematic design variables and instance parameters, updating these automatically during the flow. There are two main points of optimisation in the AutoSIF CAD tool. The first is at the wave filter coefficient calculation stage, where the optimisation is performed automatically. The second takes place after the initial memory cell design, where the user can choose to run DC and time domain simulations and manually feed results back into the tool, which optimises the initial design based upon them. In this case, the manual break in the optimisation loop is useful since it offers a designer insight into the SI design process, and the opportunity to intervene depending on their experience. Since the underlying design procedure for SI wave filters has already been presented in detail in Chapter 2, it is not repeated here, and instead the focus is on the AutoSIF cell library development (Section 3.3.1) and CAD tool implementation (Sections 3.3.2 to 3.3.6). Further details of the cell library can be found in Appendix C.

3.3.1 Cell library

Figure 3.3 illustrates the AutoSIF hierarchical analogue cell library, facilitating the design of the SI wave filters covered in Chapter 2. Beneath the top-level wave filter core symbol is a passive prototype ladder filter and a block level wave structure consisting of adaptor blocks and delay cells. If fast simulation times are required, then the hierarchy stops at level 3, where ideal behavioural models are provided for both the adaptor blocks and the delay cells. These behavioural models are either schematics made from ideal sub cells or VerilogA models.

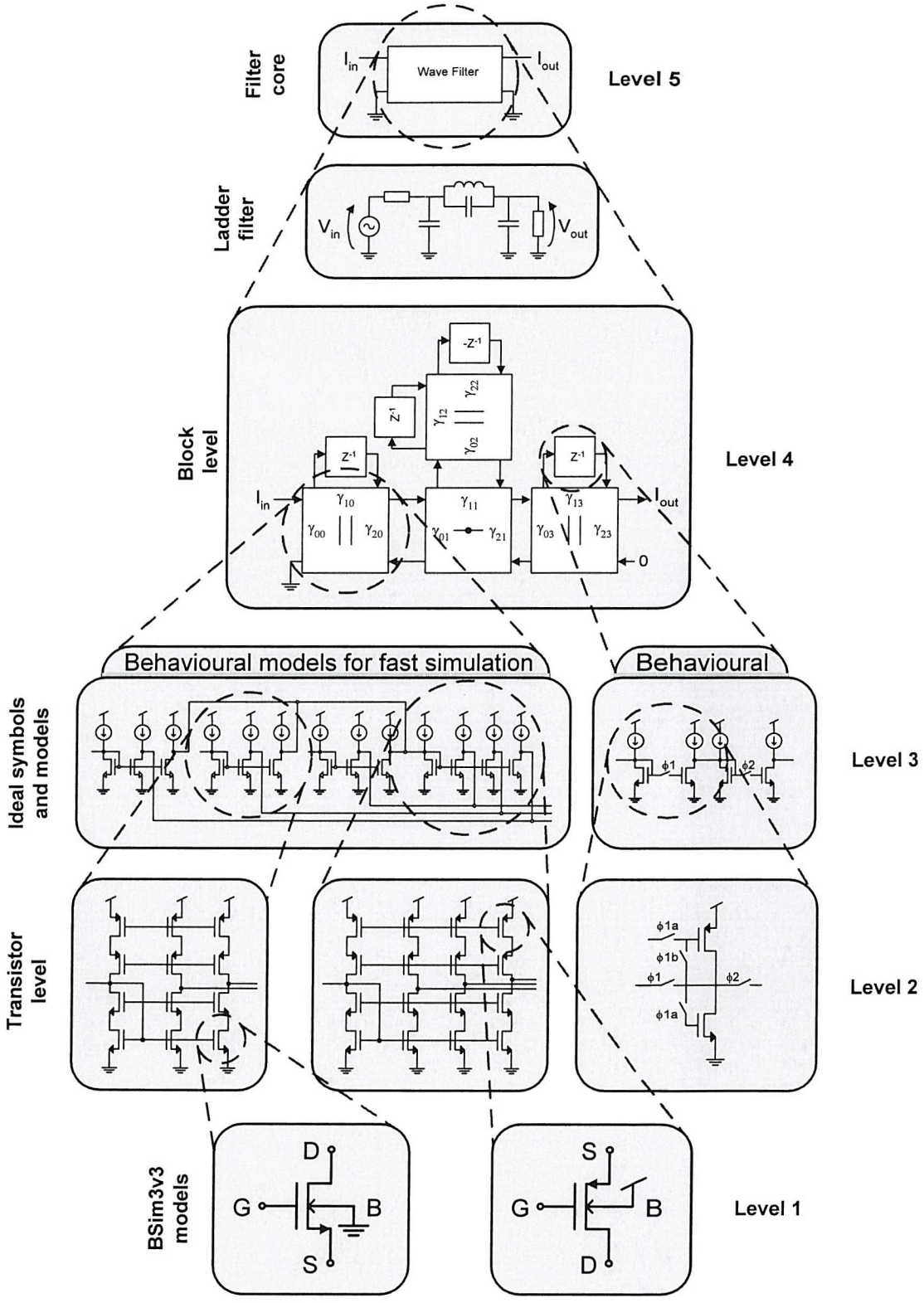


Figure 3.3 Cell hierarchy required for the design of SI wave analogue filters

If accurate simulation is required to validate a first cut design, then level 3 serves as a symbolic transistor level, and only beneath the symbolic current mirror and delay cell

symbols the true transistor level resides at level 2. In the case of Figure 3.3 the current mirror symbol has a high compliance current mirror beneath it and a positive delay cell has two cascaded S^2I memory cell beneath it. The flexibility of this approach now becomes clear since low level implementations of current mirrors and memory cells can be easily swapped in and out of the design for particular applications. Three terminal transistor symbols in level 2 hide the CMOS foundry library primitives which are at level 1. This allows easy porting of the library to a different process, since only a single NMOS and PMOS device in the NMOS and PMOS bottom level schematic need to be changed.

Transistor parameters are passed up through the hierarchy using a combination of nominal, globally defined dimensions and higher-level instance parameters, as discussed in Section 3.2.2. For example, once a number of nominal current mirror dimensions are defined, setting high level γ_{ij} coefficients in the adaptor instance properties at level 4 sets every low level transistor value in the high compliance current mirrors.

Details for a wide selection of the analogue cells in the AutoSIF wave filter core library can be found in Appendix C. A number of conventions have been followed when naming the library cellviews. A cell name ending with '`V`' denotes that this is an ideal cell whereas '`R`' denotes a real transistor level block. A cell name ending with an '`S`' indicates a symbol which could have either ideal or real levels below it. Naming conventions for filter cells consist of two parts, the first is either '`Pass`' for passive, '`Wave`' for wave ideal or '`Comp`' for transistor level with the second part reflecting the filter type function and order. A number of the AutoSIF cell library schematics are now shown to illustrate the cell hierarchy and collection of parameters and variables which must be passed between levels. Figure 3.4 shows the cellview '`CompElLo3`' which is a high level circuit structure for an elliptic lowpass 3rd order filter. In this case, the prefix '`Comp`' indicates that the hierarchy beneath the filter structure extends all the way to transistor level.

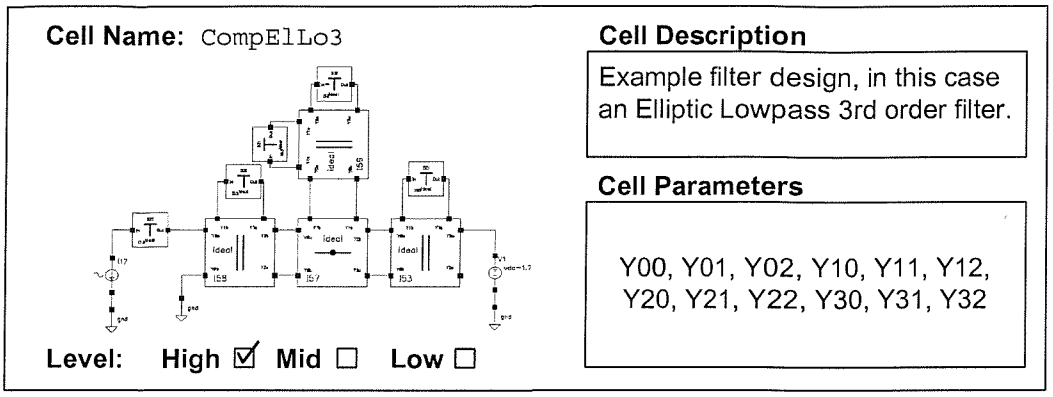


Figure 3.4 High level adaptor cell

The cell parameters include all adaptor coefficients, which are passed as parameters to the next level. Considering one of the parallel adaptors in the circuit, the next level down the hierarchy is shown in Figure 3.5. This cell view, 'ParallelR' contains current mirror symbols which hide the transistor level current mirror implementation, allowing the underlying topology to be altered if necessary. The parameters passed up the hierarchy are the three adaptor coefficients, Y0, Y1 and Y2.

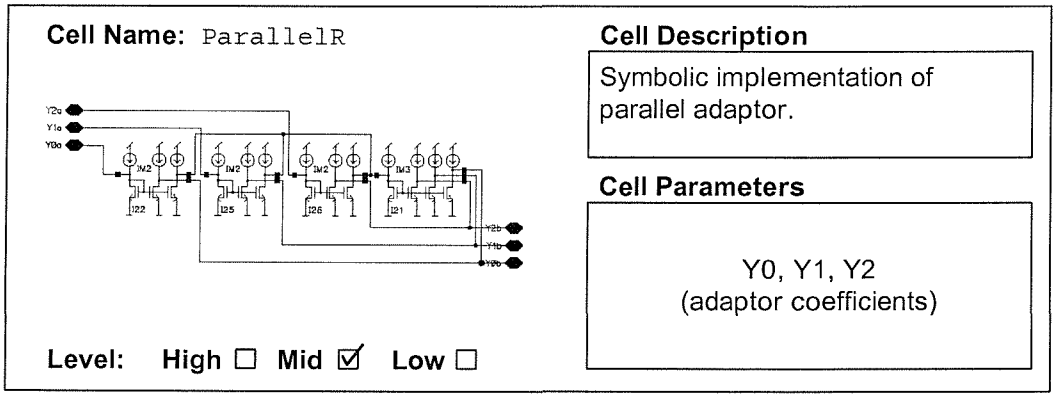


Figure 3.5 Mid level schematic

One of the current mirrors from the cellview of Figure 3.5, a three output high compliance current mirror is shown in Figure 3.6. This cellview, 'IM3R' has a large number of transistor dimensions which have to be passed up to the level above. The current mirror symbols in 'ParallelR' have instance properties for every one of the dimensions in 'IM3R', and these are set by a combination of global nominal dimensions and adaptor coefficient scaling parameters. Dummy transistors are

provided for the purpose of layout versus schematic verification when the full schematic is taken to silicon.

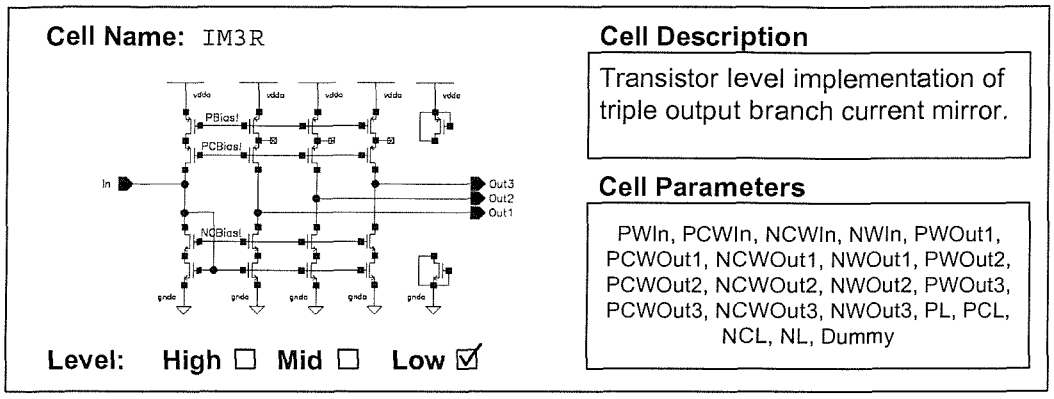


Figure 3.6 Low level current mirror

3.3.2 CAD tool overview

The AutoSIF CAD tool has been written in SKILL® to allow its integration into Cadence. Although this has a number of benefits including easy access and control of the AutoSIF cell library and use of powerful Cadence simulation tools it can unfortunately be quite a challenge to use. Apart from proprietary Cadence manuals, no extra documentation exists concerning the SKILL® language, a problem compounded by having a Cadence license for academic use only, which does not allow access to the comprehensive Cadence knowledgebase known as SourceLink.

The AutoSIF design flow consists of four steps, as shown in Figure 3.7 and follows the CAD tool methodology outlined in Section 3.2.3, consisting of ideal design and optimisation, followed by transistor level block design and overall verification. The ideal design (stage 1 of Figure 3.2) consists of both prototype filter design and ideal wave filter design and so is split into two steps, step 1 and step 2. Step 1 involves specifying and denormalising the passive filter design and during step 2 the tool automates the calculation and optimisation of wave coefficients. Step 3 involves the design of the S^2I memory cell, allowing optimisation of an initial design based on simulation results manually fed back into the tool. Finally, step 4 involves implementing the complete filter, automating the power aware scaling procedure, and verifying operation at full transistor level. All high and low level schematic

parameters and design variables are managed by the tool, and updated after the relevant design step. It should be noted that the current mirrors required for the wave filter adaptor blocks are assumed to be selected from existing libraries, and their design is not considered as part of AutoSIF.

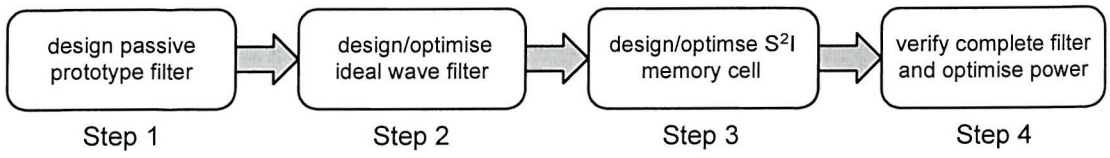


Figure 3.7 AutoSIF CAD tool design flow

Two complex data structures are used to consolidate the information used in the tool and graphical interface. The data structure `FlowDS` manages all the tool variables, arrays and lists for the internal tool operation and the data structure `FlowFm` manages all the form fields, values and components for the user interface. `FlowDS` is split into four main sub-structures: `FlowDS->Pass`, `FlowDS->Wave`, `FlowDS->S2I` and `FlowDS->Complete`, which contain all the data types necessary for the procedures associated with the four parts of the design flow. A number of other sub-structures are also part of `FlowDS` for example a collection of process parameters in `FlowDS->Process`, which are provided in a separate technology file for ease of process retargeting. Listing 3.1 shows a fragment of a process technology file. The `FlowFm` data structure also contains separate sub-structures for each of the four steps of the graphical interface. Field generation for the user interface is best achieved by maintaining a field list, for example `FlowDS->Wave->Fields` and a corresponding delete field list, for example `FlowDS->Wave->DelFields`, which allows the fields to be added and removed easily.

Listing 3.1 Process definition file

```
1     FlowDS->Process->Betan = 77.0
2     FlowDS->Process->Cgu = 2.10
3     FlowDS->Process->Cdu = 0.34
4     FlowDS->Process->Gsu = 100.0
etc…
```

The four steps of the AutoSIF tool are now considered detail. Fragments of code are given where necessary, although these only represent a fraction of the code written. In the following Sections, 3.3.3 to 3.3.6, the design flow for each step is shown using interface flow diagrams, a key for which is shown in Figure 3.8. Numbers on these flow diagrams indicate important procedure calls associated with that step.

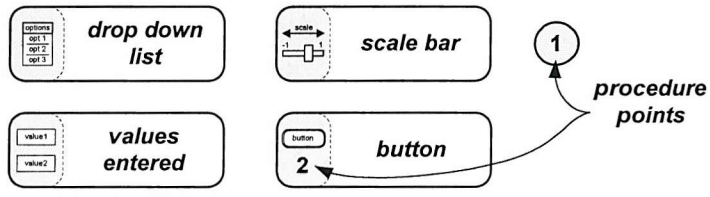


Figure 3.8 Key for interface flow diagrams

Two options exist from the main Cadence tools menu: create a new SI filter or load an existing SI filter, and it is from here that the design process begins.

3.3.3 Step 1: passive prototype filter design

The first step of the AutoSIF CAD tool involves the design of a passive prototype filter and the corresponding interface flow diagram is shown in Figure 3.9. If a new SI filter design is required, a choice of filter type, function and order is made using selection lists. If an existing design is loaded then this selection is skipped. Procedure point 1 in Figure 3.9 involves checking that the chosen filter is supported by the tool, and then loading a filter structure file which defines the passive prototype and wave filter structure. At this point AutoSIF generates the correct number of entry fields for the normalised values of the passive prototype structure. Listing 3.2 shows an example code fragment which generates the correct number of fields for the normalised capacitances. The variable `Passives` is initialised to `FlowDS->Pass->Cap`, a list containing the capacitor positions in the passive prototype filter. Lines 3-18 implement a while loop which cycles through the `Passive` list (Line 3), generating a field of the correct name (Line 8), prompt (Line 9), callback (Line 10) and default value (Line 11) and placing the field at the correct location (Line 13). Each field is added to the field list, `FlowDS->Pass->PFields` (Line 4). The while loop ends when no entries remain in `Passives`.

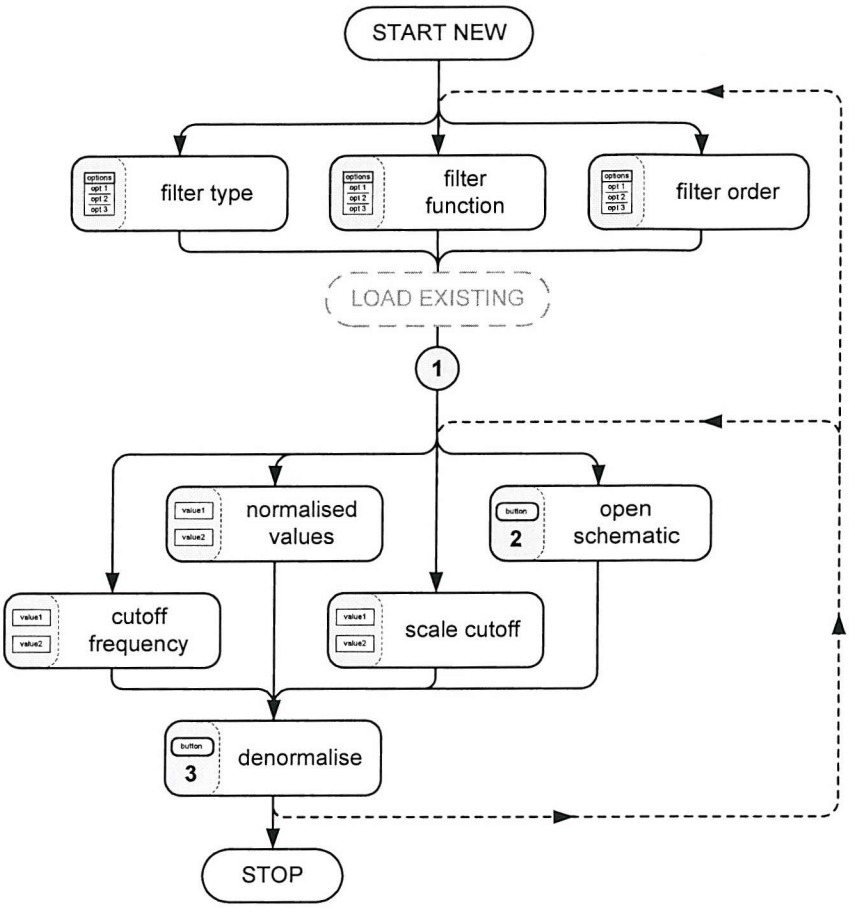


Figure 3.9 Step 1 of the AutoSIF CAD tool

Listing 3.2 Dynamic field generation (procedure point 1)

```
1     i = 0
2     Passives = cddr(FlowDS->Pass->Cap)
3     while(car(Passives)
4        FlowDS->Pass->PFields = tconc(
5        FlowDS->Pass->PFields
6        list(
7           hiCreateFloatField(
8              ?name concat('PassNC car(Passives))
9              ?prompt sprintf(nil "C%d" car(Passives))
10             ?callback sprintf(nil "FlowDS->Pass->NC[%d] =
                  FlowFm->PassNC%d->value" car(Passives) car(Passives))
11             ?defValue FlowDS->Pass->NC[car(Passives)]
12             )
13             60:(257+(30*i)) 130:25 40
14          )
15       )
16       i = i+1
17       Passives = cdr(Passives)
18    )
```

The passive prototype filter schematic can now be opened by pressing the open schematic button, (procedure point 2 in Figure 3.9). Listing 3.3 shows the function calls necessary to first open the schematic from the AutoSIF cell library (Line 1),

open Analog Artist (Line 3) and open the load state dialog to allow the default or any other simulation state to be invoked (Line 7). The prototype filter normalised values can be entered into the correct fields, and the cut-off frequency specified. A field is also provided to scale the cut-off frequency if necessary.

Listing 3.3 Open schematic and load state dialog (procedure point 2)

```
1    PassWin = geOpen(?lib "AutoSIF" ?cell
        strcat("Pass" FlowDS->Flow->File) ?view "schematic" ?mode "a")
2    hiResizeWindow(PassWin list(980:235 1268:583))
3    deInstallApp(PassWin "analogArtist-Schematic")
4    PassSessionSch = sevSession(hiGetCurrentWindow()
5    hiResizeWindow(sevWindow(PassSessionSch) list(690:618 1268:989))
6    sevCopyCellViewVariables(PassSessionSch)
7    sevLoadState(PassSessionSch)
```

The denormalise button (procedure point 3 in Figure 3.9) invokes procedures to scale the normalised component values so as to give a filter bandwidth determined by the specified cut-off frequency and scale value. Listing 3.4 shows an example SKILL® fragment used for denormalising the component values. The list `Passives` is initialised to the capacitor numbers (Line 1), and a while loop considers each capacitor in the prototype filter, calculating the denormalised value, `FlowDS->Pass->DC[]` from the normalised value, `FlowDS->Pass->NC[]` and the (scaled) cut-off frequency `FlowDS->Pass->F1` (Line 3). The form is updated with these denormalised values (Line 4) and a design variable list is created (Lines 5-11).

Listing 3.4 Passive component denormalisation (procedure point 3)

```
1    Passives = cddr(FlowDS->Pass->Cap)
2    while(car(Passives)
3       FlowDS->Pass->DC[car(Passives)] =
           1e6*(FlowDS->Pass->NC[car(Passives)]/
              (2*Pi*FlowDS->Pass->F1*FlowDS->Pass->Scale))
4       evalstring(sprintf(nil "FlowFm->PassDC%d->value =
           FlowDS->Pass->DC[%d]" car(Passives) car(Passives)))
5       PassPDesList = tconc(
6          PassPDesList
7          list(
8             sprintf(nil "C%d" car(Passives))
9             sprintf(nil "%gn" FlowDS->Pass->DC[car(Passives)])
10         )
11      )
12      Passives = cdr(Passives)
13   )
```

3.3.4 Step 2: block level wave filter design

The second step, shown in Figure 3.10, involves the design of the block level wave structure in terms of ideal adaptors and delay cells such that it is equivalent to the passive prototype filter. The user specifies a sample to cut-off frequency ratio, then using the passive component values and wave equations the tool calculates nominal wave coefficients at procedure point 1. Generic procedures have been written for the purpose of calculating the adaptor coefficients of Chebyshev and elliptic wave filters. The SKILL® fragment shown in Listing 3.5 shows a small part of the procedure used for calculating the coefficients for a Chebyshev filter.

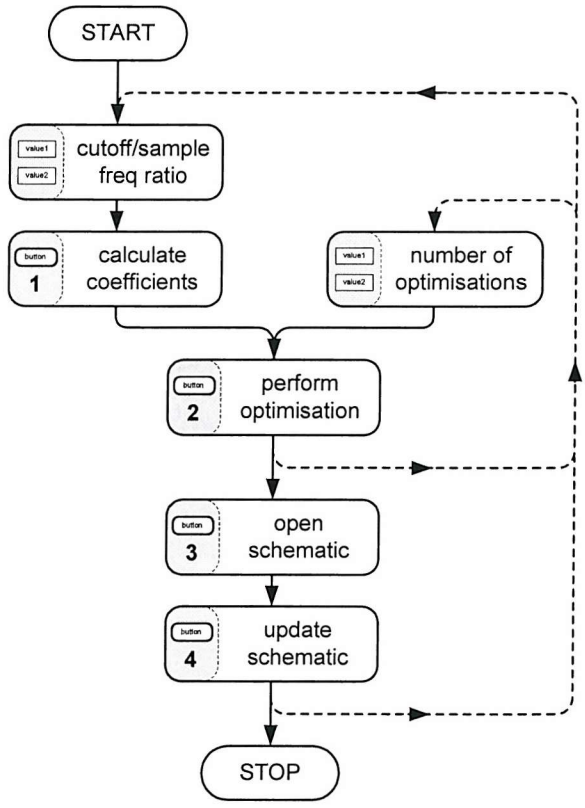


Figure 3.10 Step 2 of the AutoSIF CAD tool

First the cut-off frequency is pre-warped (Line 1) and the input and output port resistances initialised to 1.0 (Lines 2 and 3). The port 1 resistances, `FlowDS->Wave->R[i][1]`, of all the adaptors are first calculated (Lines 4-8) and once all the remaining port resistances have been determined, the adaptor coefficients, `FlowDS->Wave->Y[i][j]`, can be calculated (Lines 9-15).

Listing 3.5 Chebyshev wave coefficient calculation (procedure point 1)

```
1     FlowDS->Wave->Wp = 2*tan(Pi*FlowDS->Wave->Fr)
2     FlowDS->Wave->R[0][0] = 1.0
3     FlowDS->Wave->R[(Order-1)][2] = 1.0
4     for(i 0 Order-1
5        if(evenp(i) then
            FlowDS->Wave->R[i][1] = FlowDS->Wave->Wp/(2*
               (FlowDS->Pass->NC[(floor(i/2)+1)]/FlowDS->Pass->Scale))
6        else FlowDS->Wave->R[i][1]= (2*FlowDS->Pass->NL[(floor(i/2)+1)]/
            FlowDS->Pass->Scale)/FlowDS->Wave->Wp
7        )
8     )
etc…
9     for(i 0 Order-1
10       for(j 0 2
11          if(oddp(i) then FlowDS->Wave->Y[i][j] =
               (2*FlowDS->Wave->R[i][j])/(FlowDS->Wave->R[i][0]+
                  FlowDS->Wave->R[i][1]+FlowDS->Wave->R[i][2])
12             else FlowDS->Wave->Y[i][j] = (2*(1/FlowDS->Wave->R[i][j]))/
                  ((1/FlowDS->Wave->R[i][0])+(1/
                     FlowDS->Wave->R[i][1])+(1/FlowDS->Wave->R[i][2]))
13          )
14       )
15    )
```

Wave coefficient values will later scale transistor widths in current mirrors. As discussed in Chapter 2, wave adaptor coefficients contain a number of 'free coefficients' which are initially set to 1.0. This degree of freedom means that many different wave coefficient sets could be calculated to give exactly the same frequency response. Of these equivalent wave designs, some may exist which are particularly favourable in terms of transistor implementation. Optimisation is carried out at this stage to explore the equivalent designs taking into account two main considerations:

- ***Coefficient spread:*** Smaller transistor widths have greater sensitivity to matching and process variations than large widths. All coefficients in any one adaptor must sum to two (see Chapter 2, Section 2.2.2) and so particularly small coefficients can be avoided by ensuring that the spread of coefficient values is as small as possible.
- ***Technology grid:*** All coefficient values are implemented by transistor widths, which must lie on a standard working grid, dependent on the process being used (in our case 0.05μ). A rounding error will exist when the calculated dimension does not lie exactly on the technology grid.

The optimisation in this step of the AutoSIF CAD tool determines an optimal set of wave coefficients with respect to both of these objectives. This kind of multi-objective optimisation is particularly challenging especially if very little is known

about the dependencies of the parameters being optimised. An exhaustive search is used in the optimisation performed in this step, since the parameter space is fairly limited and the cost in terms of computation time is affordable. An algorithm has been written, based on the work in [11] to achieve the above goals and is detailed in pseudo code in Listing 3.6. The design space of possible free coefficient choices is searched, (Line 1) and for each choice, the entire coefficient set is calculated (Line 2). A sensitivity and technology grid rounding cost is generated from the coefficient set (Lines 3-6). These two costs are combined (Line 7) and if a new minimum total cost is found (Line 9) then the coefficients are considered optimal and are locally stored (Line 11). The granularity of the free coefficient space search determines the number of times the inner loop must be executed, and this parameter chosen through the 'number of steps' field on the user interface.

Listing 3.6 Optimising the wave coefficients (procedure point 2)

```
1     traverse free coefficient space (
2        calculate set of coefficients from choice of free coefficients
3        calculate sensitivity for each coefficient
4        generate sensitivity cost
5        calculate tech rounding error for each coefficient
6        generate tech cost
7        combine Sens and Tech to give TotalCost
9        if TotalCost is better than BestCost then {
10          BestCost = TotalCost
11          OptCoeffs = Coeffs
12       }
13    }
```

Feedback on the optimisation progress is also given through the Cadence Command Interface Window (CIW), which includes a percentage improvement in sensitivity and technology grid rounding errors, as compared to the initial coefficient set. Once the user is happy with the optimised coefficients, the 'open schematic' and 'update schematic' buttons (procedure points 3 and 4) can be used to open the ideal wave filter schematic for the filter being designed and then assert the wave coefficients. The SKILL® code for opening the wave filter schematic is similar to that in Listing 3.3. Listing 3.7 shows a SKILL® fragment for generating the design variable list for the optimised wave coefficients. A two dimensional loop (Lines 1-2) concatenates the value of every adaptor coefficient to the design variable list (Lines 3-9). The design variables are updated using the appropriate function (Line 12) and the frequency response of the ideal wave filter can then be simulated using SepctreRF.

Listing 3.7 Creating a design variable list (procedure point 4)

```
1     for(i 0 (car(FlowDS->Wave->Series)+car(FlowDS->Wave->Parallel)-1)
2        for(j 0 2
3           WaveDesList = tconc(
4              WaveDesList
5              list(
6                 sprintf(nil "Y%d%d" i j)
7                 sprintf(nil "%g" FlowDS->Wave->OY[i][j])
8              )
9           )
10       )
11    )
12    asiSetDesignVarList(asiGetSession(WaveWin) car(WaveDesList))
```

3.3.5 Step 3: memory cell design

The complete memory cell design process is detailed in Chapter 2, Section 2.4.1 and is implemented during step 3 of the AutoSIF CAD tool, shown in Figure 3.11. Five main scale fields drive the first cut design, including sampling frequency, modulation index and bias current. As these values are changed, the first cut design fields are dynamically calculated (procedure point 1). If the user specifications would cause the memory cell transistors to exit the saturation region, all of the first cut design fields become shaded grey, indicating an unfeasible design. Procedure point 1 is simply an automated series of calculations, following the design process of the S^2I memory cell given in Chapter 2, Section 2.4.1. A SKILL® fragment shown in Listing 3.8 demonstrates a small number of the calculations required to generate the first cut dimensions. First, cell parameters such as the memory transistor transconductance (Line 1) and total summing node capacitance (Line 2) are calculated, and then the cell dimensions, for example the NMOS width (Line 3) and length (Line 4) are calculated and these are subsequently rounded to the technology grid (Lines 5-6).

Listing 3.8 Calculating memory cell parameters (procedure point 1)

```
1     FlowDS->S2I->Gm = ((2*FlowDS->S2I->Bias)/FlowDS->S2I->Vgt)
2     FlowDS->S2I->Ctot = ((1000/(12*FlowDS->S2I->Fs))*FlowDS->S2I->Gm)
etc…
3     FlowDS->S2I->Wn = sqrt( FlowDS->S2I->Area * FlowDS->S2I->WnLn )
4     FlowDS->S2I->Ln = FlowDS->S2I->Wn / FlowDS->S2I->WnLn
etc…
5     FlowDS->S2I->Wn = FlowDS->Process->TechGrid*
         round(FlowDS->S2I->Wn/FlowDS->Process->TechGrid)
6     FlowDS->S2I->Ln = FlowDS->Process->TechGrid*
         round(FlowDS->S2I->Ln/FlowDS->Process->TechGrid)
```

The first cut fields in the interface are then automatically updated, and a number of performance fields are also calculated, for example the filter power, and cell SNR. This step allows a designer to explore the SI memory cell design space, observing the direct influence of parameters on the cell performance.

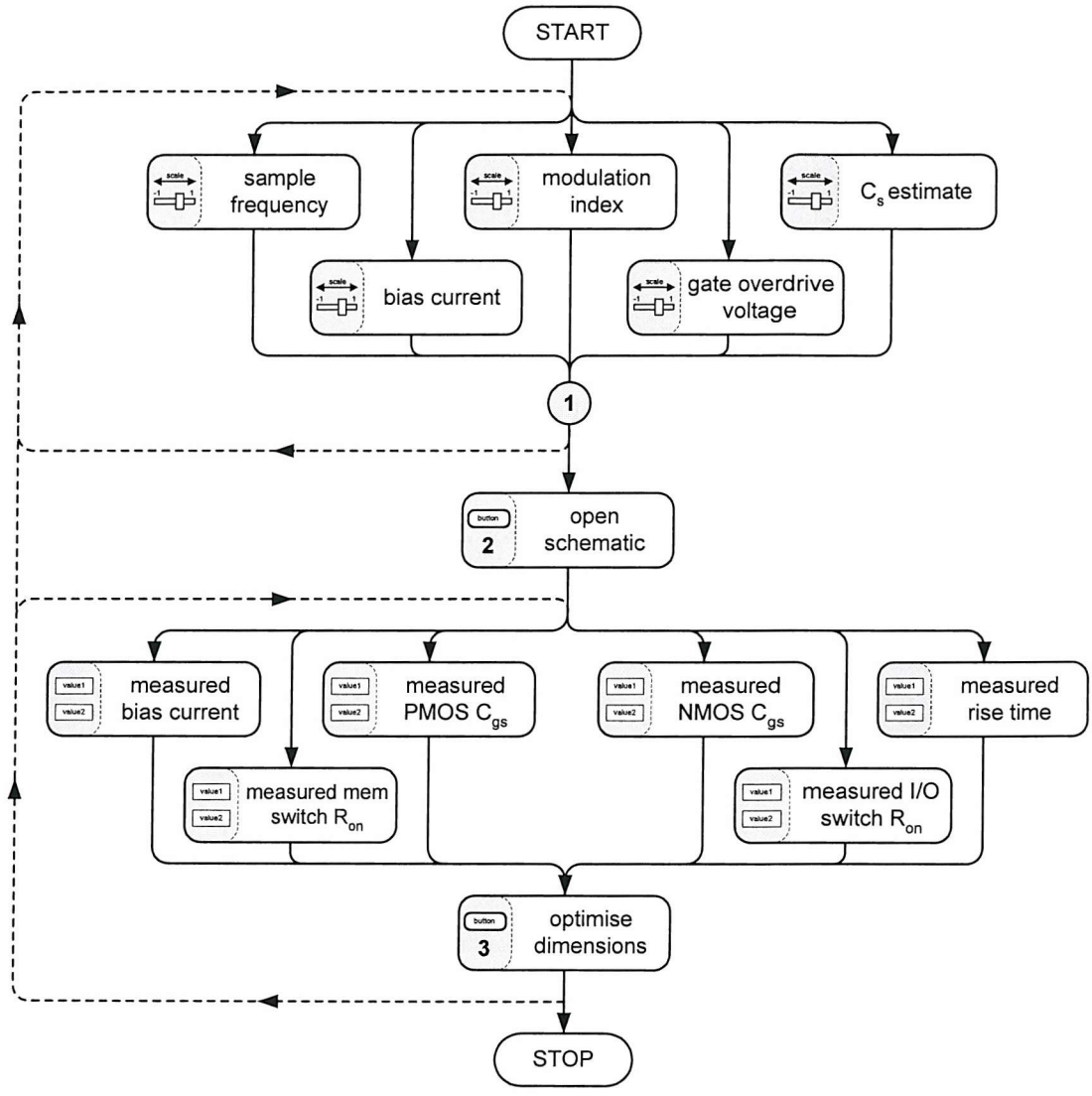


Figure 3.11 Step 3 of the AutoSIF CAD tool

Once a first cut design has been reached, the schematic can be opened by pressing the open schematic button (procedure point 2). Similar to AutoSIF steps 1 and 2, this opens a schematic containing test structures and loads a default state dialog (see Listing 3.3). Following Bsim3v3 simulations, operating point and time domain values are manually fed back into the tool into the six measured values fields (Figure 3.11), including the measured memory cell drain currents I_D', gate capacitances, C_{gp}'

and C_{gn}', and switch resistances R_{si}' and R_{sm}'. The 'optimise dimensions' button invokes simple optimisation of the first cut transistor dimensions based on these measured results (procedure point 3). Gate capacitances, C_{gp} and C_{gn}, are always designed to be equal yet both these and the drain current, I_D are affected by altering the memory cell transistor dimensions, w_p/l_p and w_n/l_n and so it is important to satisfy either of the optimisations without affecting the other. Eq. (3.1) and (3.2) use the measured drain current, I_D' to derive, $(w_n,w_p)_{new}$ and $(l_n,l_p)_{new}$ such that the drain current is much closer to the designed I_D without affecting gate capacitances. Likewise Eq. (3.3) and (3.4) use the measured gate capacitances, C_{gp}' and C_{gn}' to make the second iterations closer to the designed gate capacitance, C_{gs}. Listing 3.9 shows an example SKILL® fragment for one optimisation step, equivalent to Eq. (3.2). First the new optimised dimensions are calculated (Lines 1-2) and these are then rounded to the technology grid (Lines 3-4). The optimised dimensions are shown at the bottom of the user interface.

$$(w_n, w_p)_{new} = \sqrt{\frac{I_D}{I_D'}} \cdot (w_n, w_p)_{old} \qquad (3.1)$$

$$(l_n, l_p)_{new} = \sqrt{\frac{I_D'}{I_D}} \cdot (l_n, l_p)_{old} \qquad (3.2)$$

$$(w_n, l_n)_{new} = \sqrt{\frac{C_g}{C_{gp}'}} \cdot (w_n, l_n)_{old} \qquad (3.3)$$

$$(w_p, l_p)_{new} = \sqrt{\frac{C_g}{C_{gn}'}} \cdot (w_p, l_p)_{old} \qquad (3.4)$$

Listing 3.9 Optimising memory cell parameters (procedure point 3)

```
1     FlowDS->S2I->Ln = (FlowDS->S2I->Ln/((FlowDS->S2I->Bias/
         FlowFm->MemMesBias->value ) ** 0.5 ) )
2     FlowDS->S2I->Lp = ( FlowDS->S2I->Lp / ( ( FlowDS->S2I->Bias /
         FlowFm->MemMesBias->value ) ** 0.5 ) )
etc…
3     FlowDS->S2I->Wn = FlowDS->Process->TechGrid*
         round(FlowDS->S2I->Wn/FlowDS->Process->TechGrid)
4     FlowDS->S2I->Ln = FlowDS->Process->TechGrid*
         round(FlowDS->S2I->Ln/FlowDS->Process->TechGrid)
```

Input/output switches are optimised in the same way, but for their on-resistance. The memory switch width can also be optimised, and this is provided for in step 3, however, a better method of optimisation is to run a parametric simulation to obtain a family of drain current curves for different switch widths. The curve and hence width which gives the most critically damped settling can then be determined.

3.3.6 Step 4: transistor level filter verification

Step 4, shown in Figure 3.12 involves replacing the ideal blocks of step 2 with the transistor level blocks designed in step 3. Furthermore, the ideal wave filter design of step 2 is used to determine current mirror peak values which facilitate the two part power-aware scaling method outlined in Chapter 2, Section 2.3. The current mirrors of the adaptor blocks are picked from pre-designed highly optimised cells, and as such are not designed as part of the design flow. The current mirrors used at present are of the high-compliance cascoded structure, detailed in Chapter 2, Section 2.4.2.

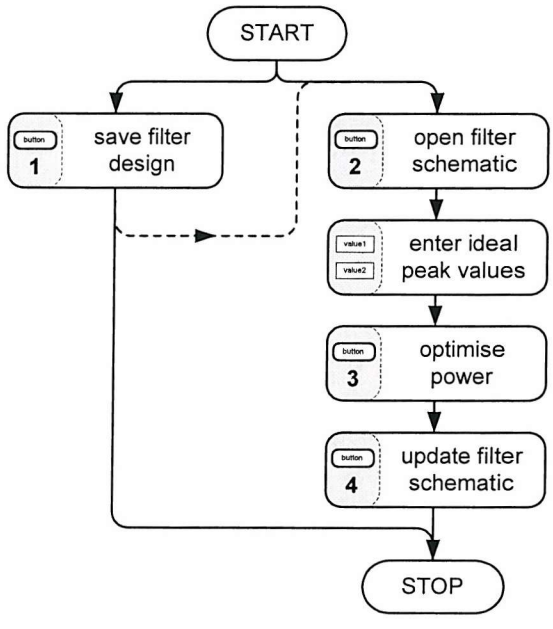


Figure 3.12 Step 4 of the AutoSIF CAD tool

At any point during or before step 4 it is possible to save the entire filter state to allow the design to be continued at a later stage (procedure point 1). The entire data structure `FlowDS` is disassembled into a save state file of a given name. To facilitate this, a number of procedures have been written to allow 1 or 2 dimensional arrays to

be stored and restored to and from a single list. Listing 3.10 shows a SKILL® fragment illustrating the principle of the saving process, where two arrays are converted to a list (Line 1-2), a filename prompt is generated (Line 3), an output printing port is created (Line 4) and then the decomposed data structure, `FlowDS` is written to the specified save state file.

Listing 3.10 Saving the data structure (procedure point 1)

```
1     FlowDS->Pass->NC = FlowStore1DArray(FlowDS->Pass->NC)
etc…
2     FlowDS->Wave->Y = FlowStore2DArray(FlowDS->Wave->Y)
etc…
3     FlowDS->Flow->SaveFile =
         FlowGetFileName("Save state filename:" FlowDS->Flow->SaveFile)
4     prt = outfile(strcat(SIFilterStatesPath
         FlowDS->Flow->SaveFile ".state"))
5     pprint(FlowDS prt)
6     close(prt)
```

The SKILL® fragment in Listing 3.11 shows the principle behind restoring the saved state. First a printing port is opened (Line 1) and then the data structure `FlowDS` is filled with the values in the file (Line 2). The collapsed arrays are then restored (Lines 4-5) and the fields regenerated (Lines 6-9).

Listing 3.11 Restoring the saved state

```
1     prt = infile(strcat(SIFilterStatesPath FileName ".state"))
2     FlowDS = car(lineread(prt))
3     close(prt)
4     FlowDS->Pass->NC = FlowRestore1DArray(FlowDS->Pass->NC)
etc…
5     FlowDS->Wave->Y = FlowRestore2DArray(FlowDS->Wave->Y)
etc…
6     PassMakeFields()
7     WaveMakeFields()
8     S2IMakeFields()
9     CompleteMakeFields()
```

A schematic for the entire transistor level filter can be opened and the Cadence simulation environment started at this stage by pressing the open schematic button (procedure point 2) which uses a procedure similar to Listing 3.3. The schematic appears similar to the ideal wave schematic of step 2, but actually contains a full hierarchy, all the way to the low level transistors and furthermore contains the clock and bias structures necessary for correct filter operation.

An important step remains to complete the filter design, namely the power-aware bias and signal scaling method (procedure point 3). Since this step is discussed in detail in Chapter 2, Section 2.3 it is not repeated to a similar complexity here. The pseudo code in Listing 3.12 shows the procedure used to scale all the current mirror branch widths to prevent non-homogeneous signal levels and distortion and reduce power. Three lists are generated, with one entry for each of the current mirror branches in every current mirror in the entire filter. It should be noted that there is more support for list data types in the SKILL® language than for data arrays, and for this reason it is common to use multi dimensional lists as an alternative to implementing arrays. The first list (Line 1) puts a 1.0 in every current mirror branch position other than those branches which must implement the adaptor coefficients, where the correct adaptor coefficient is placed instead. The second list (Line 2) uses the peak values read from the input branches of every current mirror to determine scaling constants for all the branches in that current mirror, and fills the list accordingly. The last list (Line 3) uses the peak values from the delay cells to generate scaling values for attached current mirror branches. The three lists are then multiplied together (Line 4) to determine a scale value for every branch in every current mirror. The complete set of scaling values simultaneously implements adaptor coefficients, and the scaling of bias currents and signal levels in the filter to prevent distortion and reduce power consumption.

Listing 3.12 Power aware scaling (procedure point 3)

```
1    generate a list of the adaptor coefficient contributions
2    generate a list of the adaptor scaling values
3    generate a list for the delay scaling values
4    multiply the above three lists together
5    use the resulting total scale values to scale every current mirror
```

The overall power aware scaling procedure is implemented in over 600 lines of SKILL® and a detailed consideration of this implementation is not given here. However, Listing 3.13 does show the SKILL® code required to generate the list of the adaptor coefficient scaling values (line 1, Listing 3.12), which is the simplest of the three scaling contributions. The procedure considers every adaptor block in turn (Line 3) and after determining whether it is a series or parallel adaptor (Line 4) appends the correct collection of scaling values to the list (Lines 7 or 12). It can be

seen that the optimised wave coefficients `FlowDS->Wave->OY[][]` are implemented as scaling factors in the correct branches for parallel (Lines 8-10) and series (Line 16) adaptors.

Listing 3.13 Creating the adaptor coefficient list.

```
1     j = 0
2     List = nil
3     for(i 0 car(FlowDS->Wave->Series)+car(FlowDS->Wave->Parallel)-1
4        if((car(nthcdr((j+1) FlowDS->Wave->Parallel))==i)
5           then
6           j = j+1
7           List=append(List
8              list(list(list(1.0 1.0 FlowDS->Wave->OY[i][0])
9                 list(1.0 1.0 FlowDS->Wave->OY[i][1])
10                list(1.0 1.0 FlowDS->Wave->OY[i][2])
11                list(1.0 1.0 1.0 1.0))))
12          else List = append(List
13                list(list(list(1.0 1.0 1.0)
14                list(1.0 1.0 1.0)
15                list(1.0 1.0 1.0)
16                list(1.0 FlowDS->Wave->OY[i][0] FlowDS->Wave->OY[i][1]
                        FlowDS->Wave->OY[i][2])
17                list(1.0 1.0)
18                list(1.0 1.0)
19                list(1.0 1.0))))
20       )
21    )
```

Design variables can be asserted in the schematic by pressing the update schematic button (procedure point 4), which then allows full transistor level filter simulations to be run using the SpectreRF switched circuit simulator. After verification of the full transistor level filter it is likely that the next stage of the filter design is layout. Layout is not included as part of the design process for a number of reasons, not least because this would tie the entire flow to a particular technology but also because in industry, analogue layout is normally undertaken by experienced in-house design engineers using company standard techniques. The tool does, however, provide a good aid for layout, in the form of generating a '.dims' text output file which specifies the entire set of transistor dimensions, adaptor by adaptor, mirror by mirror and branch by branch. An extract from a typical file is shown in Listing 3.14, where unlike a netlist for example, the output is in a more human readable form and includes useful information such as technology grid rounded values and associated errors.

Listing 3.14 Example section of a dimensions file

```
This file contains dimensions for a third order elliptic Lowpass filter


Base Mirror Dimensions are:

PW (PMOS Width):                120.00um
PCW (PMOS Cascode Width):       120.00um
NCW (NMOS Cascode Width):       40.00um
NW (NMOS Width):                40.00um
PL (PMOS Length):               5.00um
PCL (PMOS Cascode Length):      1.50um
NCL (NMOS Cascode Length):      1.50um
NL (NMOS Length):               5.00um


The current acceptable percent implementation error is: 0.10%
(Violations of this limit are flagged with three stars (***)


Adaptor Number 0:

Dimension:  Exact:      Rounded:      Error:
M1InPW      120.000     120.00        0.0000%
M1InPCW     120.000     120.00        0.0000%
M1InNCW     40.000      40.00         0.0000%
M1InNW      40.000      40.00         0.0000%
M1Out1PW    120.000     120.00        0.0000%
M1Out1PCW   120.000     120.00        0.0000%
M1Out1NCW   40.000      40.00         0.0000%
M1Out1NW    40.000      40.00         0.0000%
M1Out2PW    33.047      33.05         0.0080%
M1Out2PCW   33.047      33.05         0.0080%
M1Out2NCW   11.016      11.00         0.1433% ***
M1Out2NW    11.016      11.00         0.1433% ***
M2InPW      120.000     120.00        0.0000%
M2InPCW     120.000     120.00        0.0000%
M2InNCW     40.000      40.00         0.0000%
M2InNW      40.000      40.00         0.0000%
M2Out1PW    120.000     120.00        0.0000%
M2Out1PCW   120.000     120.00        0.0000%
M2Out1NCW   40.000      40.00         0.0000%
M2Out1NW    40.000      40.00         0.0000%
M2Out2PW    146.953     146.95        0.0018%
M2Out2PCW   146.953     146.95        0.0018%
M2Out2NCW   48.984      49.00         0.0323%
M2Out2NW    48.984      49.00         0.0323%
M3InPW      120.000     120.00        0.0000%
M3InPCW     120.000     120.00        0.0000%
M3InNCW     40.000      40.00         0.0000%
M3InNW      40.000      40.00         0.0000%
M3Out1PW    120.000     120.00        0.0000%
M3Out1PCW   120.000     120.00        0.0000%
M3Out1NCW   40.000      40.00         0.0000%
M3Out1NW    40.000      40.00         0.0000%
M3Out2PW    60.000      60.00         0.0000%
M3Out2PCW   60.000      60.00         0.0000%
M3Out2NCW   20.000      20.00         0.0000%
M3Out2NW    20.000      20.00         0.0000%
M4InPW      120.000     120.00        0.0000%
M4InPCW     120.000     120.00        0.0000%
M4InNCW     40.000      40.00         0.0000%
M4InNW      40.000      40.00         0.0000%

etc…
```

3.4 Application example

Details of the wave filter design process have been described in Chapter 2, Sections 2.2-2.4 and only a brief example is given here to illustrate the use of AutoSIF. The example used is a 5th order elliptic lowpass filter of 1kHz cut-off, 35dB stopband attenuation and <100mW power consumption. Figure 3.13 shows step 1 of the interface where the filter type, function and order has been chosen and suitable normalised values entered. These values are then denormalised to a cut-off of 1kHz and the schematic updated. An AC simulation is then run to determine correct operation of the passive design and the simulated response is shown in Figure 3.14.

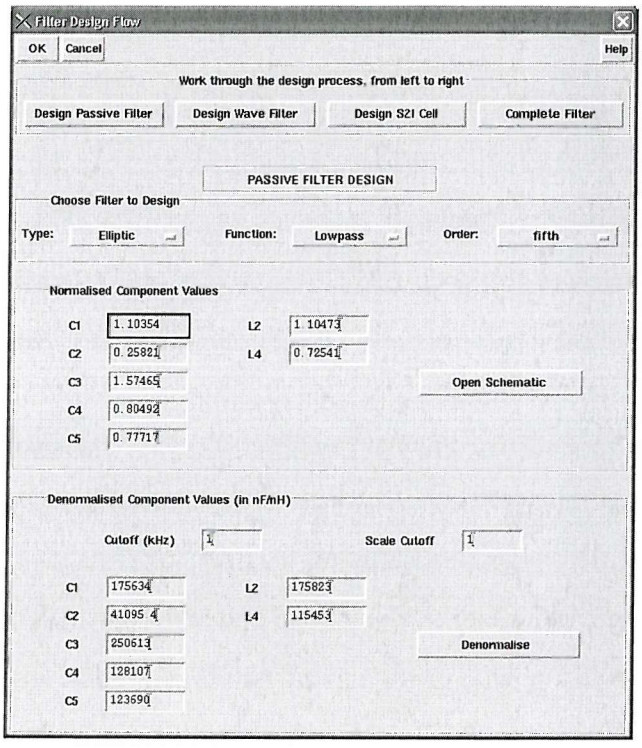


Figure 3.13 Step 1: designing the passive reference filter

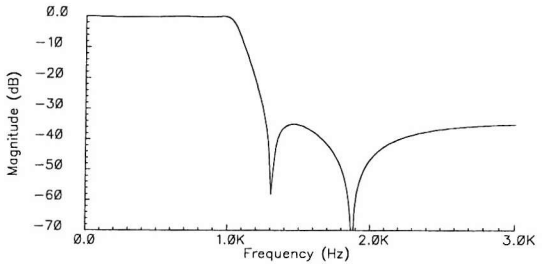


Figure 3.14 Passive reference filter response from step 1

Step 2 of the design flow is shown in Figure 3.15 where a cut-off to sampling frequency ratio of 0.1 has been specified. The initial coefficients are calculated with the free coefficients (in this case γ_{02}, γ_{12}, γ_{20}, γ_{32}, γ_{42}, γ_{50}) set arbitrarily to 1. The set of coefficients are then optimised which has raised the smallest value from 0.198 to 0.255 hence improving the sensitivity of the design. The ideal wave schematic is opened and updated and the ideal wave response can then be determined.

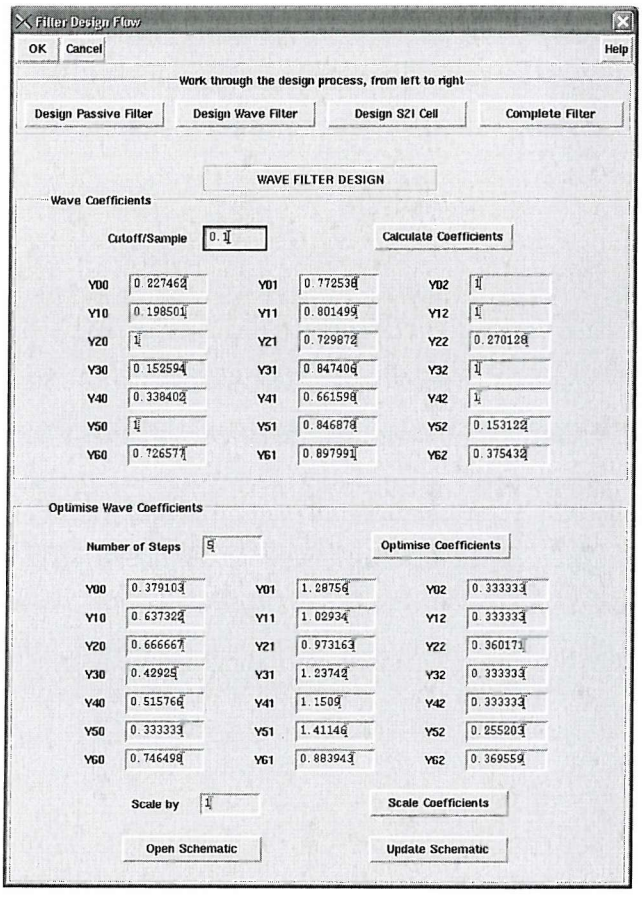


Figure 3.15 Step 2: calculating and optimising the wave filter coefficients

Step 3 of the design flow is shown in Figure 3.16 where a sampling frequency of 1MHz is chosen for the design (this is more than sufficient for the 10kHz sampling frequency of this design). The bias current is chosen as 100mA which gives a power consumption well within the 100mW budget, and the first cut design calculated. A schematic containing test structures is opened and DC and transient simulations are used to optimise the first cut design, giving the improved dimensions at the bottom of the form.

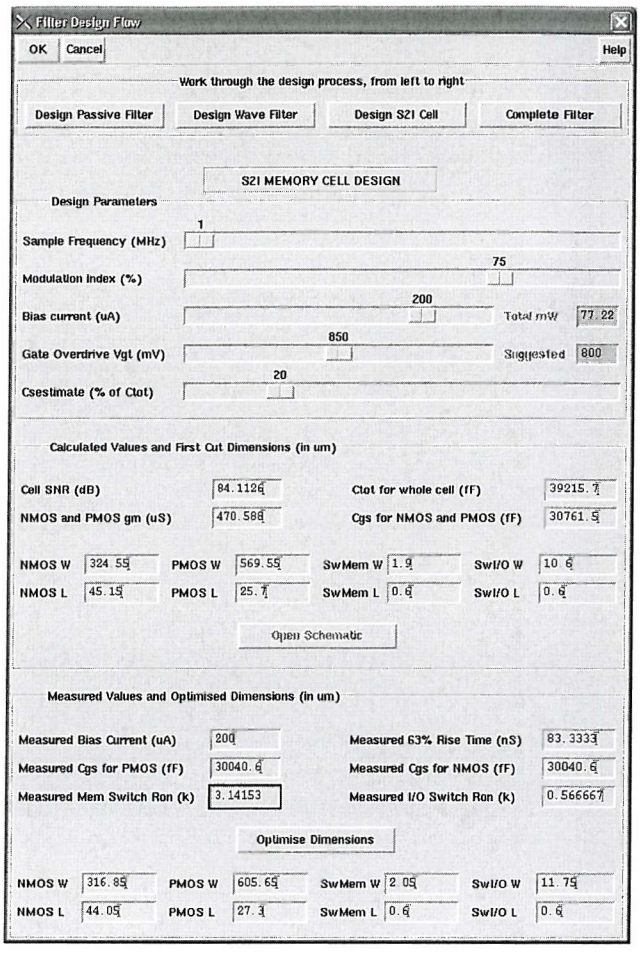


Figure 3.16 Step 3: designing and optimising the S^2I cell

The final step of the flow involves using the ideal wave filter simulation of step 2 to determine the peak of the frequency response for every input node to the current mirrors and delay cells in the filter. These values are filled in the form which then calculates and combines a number of sets of scaling values to produce an entire set of scaling factors for the filter. The full transistor level filter is opened and these dimensions are asserted. Simulations can then be run at this level. The full transistor level simulation for the example 5^{th} order elliptic filter is shown in Figure 3.18(a) where a close agreement to the wave ideal response can be seen. The design process has also been used to also design another 5^{th} order elliptic filter, but in this case to a larger stopband attenuation specification of >60dB. The response for this second filter can be seen in Figure 3.18(b). Both filters took little more than an hour to design from start to finish. The memory cell designed in step 3 is suitable for up to a

1MHz sampling frequency and this is shown in Figure 3.19 which shows the response of the second filter when sampled at 1MHz, giving a cut-off of 100kHz.

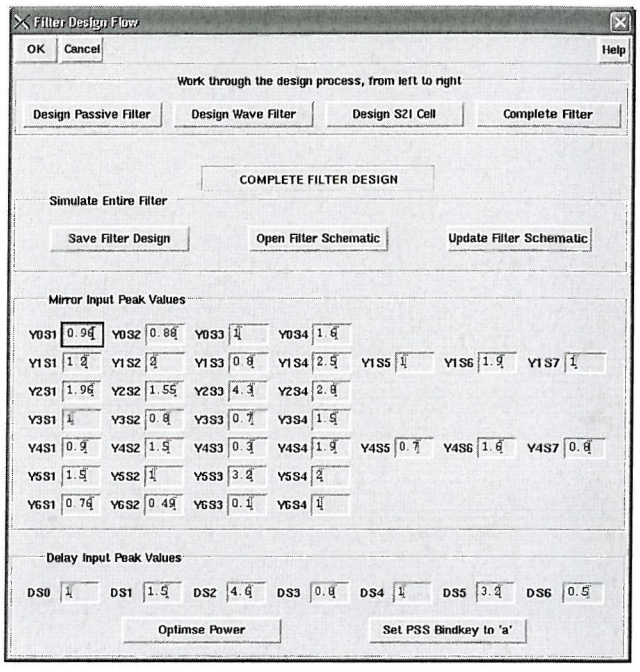


Figure 3.17 Step 4: optimising for power and verifying complete filter

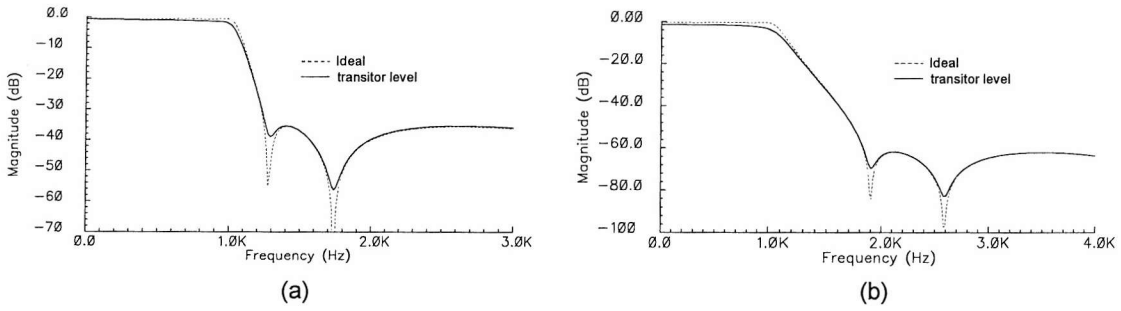


Figure 3.18 Transistor level response for the example (a) and a second design (b)

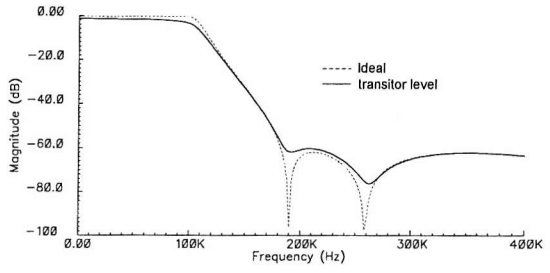


Figure 3.19 Operating the second design at a sampling frequency of 1MHz

3.5 Concluding remarks

The SI technique has the potential to solve the analogue bottleneck in SoC, however, a lack of tools and understanding is hindering its acceptance in the design world. This chapter has carefully considered key features of CAD tools proposed in the literature to derive a novel two part CAD methodology most suited to analogue IP core generation. The methodology, discussed in Section 3.2 comprises a carefully developed and parameterised analogue cell library and a tool, which automates well captured design steps. A key feature of the proposed approach is that it is completely integrated within the industry standard EDA design suite, Cadence, allowing a truly seamless design process. Section 3.3 details the application of this methodology to the design of SI wave filter cores in the form of AutoSIF, a cell library and tool not only capable of rapid development of SI filter cores but also providing excellent insight into the resulting SI design. Optimisation is performed at both block level for wave filter adaptor coefficients and transistor level for low level memory cell dimensions. AutoSIF also automates the power-aware bias and signal scaling method shown in Chapter 2 to prevent distortion in wave filters whilst reducing power consumption. Example filter designs, including a 5^{th} order elliptic lowpass filter with 100kHz cut-off, are given in Section 3.4, which demonstrate the ability of AutoSIF to generate high performance filter specifications with a very fast turnaround.

Chapter 4

Switched-Current Phase-Locked Loops

Chapters 2 and 3 considered design methods and tools for switched-current (SI) filter cores, with validation through the design and fabrication of a silicon prototype. This chapter addresses the design of a second SI core. SI data converters have been reported in the literature [18-21, 112-116], however, there has been very little reported work in the area of applying the SI technique to phase-locked loop (PLL) design [24]. PLLs are fundamental building blocks, employed in a vast range of important applications spanning from clock recovery, frequency shift keying (FSK), to demodulation and frequency synthesis. This chapter presents a novel 2^{nd} order SI PLL architecture and demonstrates its use in practical applications. Section 4.1 details existing work in the area of SI PLLs and provides an overview of conventional PLL architectures to an extent required by later sections. Section 4.2 presents the novel 2^{nd} order SI PLL architecture, describing the implementation, modelling and theoretical analysis of the major SI PLL blocks. Section 4.3 gives the transistor implementation of these main blocks illustrated through the use of a case study example. The SI PLL design flow has been automated in the form of a CAD methodology called AutoPLL which is detailed in Section 4.4. Section 4.5 presents experimental results from two PLL case studies, a FSK demodulation example based on the case study of Section 4.3 and a frequency synthesis example with specifications similar to a state of the art PLL IP core commercially available [117].

4.1 Preliminary review

4.1.1 Literature review

To the best of the author's knowledge, there exists only one example of a SI PLL in the literature, published in 1997 [24]. The PLL in [24] has a conventional all digital architecture but uses a SI integrator in place of the standard digital up/down counter. A 1st order loop is used and measured specifications report a centre frequency of 1MHz, area of 3mm^2 (on a 2.4μm process), power consumption of 1.85mW and phase jitter of 20°. PLLs utilising switched capacitor (SC) filters have been reported in the literature, for example in [35], however, the SC technique fundamentally requires integrated linear capacitors, unlike the SI approach. The main contribution of the SI PLL work in [24] is that it shows the possibility to design simple 1st order PLL circuits using transistors alone and hence without the need for linear capacitors. However, most current and future practical PLL applications require 2nd order architectures with higher frequencies of operation than 1MHz [118]. SI techniques have a number of features which may be beneficial when applied to the area of fully integrated PLL design, such as low power consumption (through the use of class AB structures), high frequency operation and reduced area. These features along with the lack of previous work in the area are key motivating factors for developing the novel SI PLL architecture presented in this chapter.

4.1.2 Conventional PLL architecture

An analysis of a conventional mixed-signal PLL architecture is provided here to an extent required by later sections, and more details can be found in a suitable standard text [118]. Figure 4.1 shows a standard mixed-signal PLL architecture which operates as follows. An input signal $\Theta_1(s)$ and a reference signal $\Theta'_2(s)$ are compared in phase by a phase detector (PD) whose output is proportional to the phase error multiplied by a phase detector gain (K_d). This signal passes through a loop filter (LF) which reduces its higher frequency content, leaving a steady value for the oscillator, which in the context of a SI PLL is a *current* controlled oscillator (ICO). The output

of the ICO $\Theta_2(s)$ has a frequency given by its centre frequency plus its frequency gain (K_o) multiplied by the input signal. A divide by N stage allows frequency synthesis if required. Given a frequency step in $\Theta_1(s)$, the phase difference between this and $\Theta'_2(s)$ will gradually change resulting in a change in the output of the PD. The DC component of the filter output will cause the ICO output frequency $\Theta_2(s)$ to adjust to minimise the phase error. At some point the filter output will settle at a new value which is sufficient to maintain the ICO output at a frequency exactly equal to N times the new input frequency. In a 2nd order loop, once the PLL is locked, there will therefore be a known static phase error between the input signal, $\Theta_1(s)$, and reference signal, $\Theta_2'(s)$.

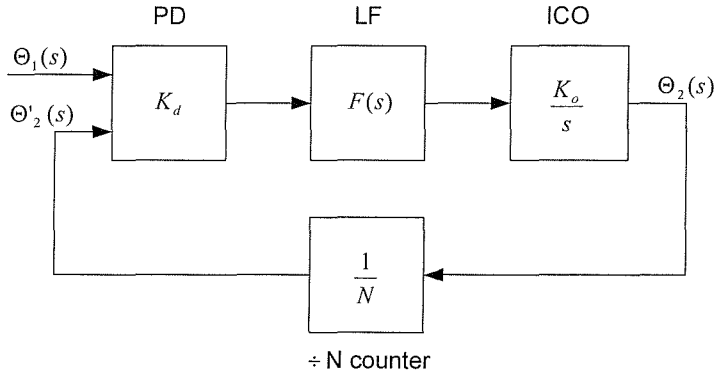


Figure 4.1 Block diagram of a generic PLL

Even the simplest PLL with no LF has a 1st order loop due to a pole in the ICO. This pole exits because by definition, phase is given by integrating over frequency, which corresponds to division by s in the Laplace domain. The Laplace transform of the ICO is therefore K_o/s [118]. A 1st order loop is fundamentally stable but the settling response can be too slow for many applications. Most practical PLLs have 2nd order loops, with one pole in the LF and one in the ICO which allows a faster settling response with a controllable trade-off between overshoot and settling time. Some PLLs are designed with loops of even higher order, allowing more poles to be placed in the LF transfer function and the response to be improved further. However, the placement of these poles is critical since too much phase at the unity gain frequency will result in the system becoming unstable, and oscillating. The SI PLL architecture presented in this chapter has a 2nd order loop with damping factor ζ and undamped

natural frequency ω_n. It is important to be able to control these parameters independently, and so commonly a lead lag LF is used which introduces a zero into the filter transfer function [119]. It can be readily shown that the phase transfer function for the PLL of Figure 4.1 is:

$$H(s) = \frac{K_d F(s) K_o / N}{s + K_d F(s) K_o / N} \tag{4.1}$$

The passive implementation of the lead lag filter used in the majority of practical PLLs is shown in Figure 4.2. and has the following transfer function:

$$F(s) = \frac{1 + s\tau_2}{1 + s(\tau_1 + \tau_2)} \tag{4.2}$$

where: $\tau_1 = R_1 C \qquad \tau_2 = R_2 C$ (4.3)

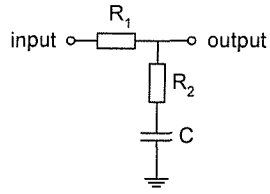


Figure 4.2 Passive lead lag filter

Substituting Eq. (4.2) into Eq. (4.1) leads to the overall loop transfer function of:

$$H(s) = \frac{\frac{K_d K_o}{N} \frac{1 + s\tau_2}{(\tau_1 + \tau_2)}}{s^2 + s\frac{1 + K_d K_o \tau_2 / N}{(\tau_1 + \tau_2)} + \frac{K_d K_o / N}{(\tau_1 + \tau_2)}} \tag{4.4}$$

The denominator of Eq. (4.4) has been purposely written in the normalised form of $s^2 + 2\zeta\omega_n s + \omega_n^2$, allowing the natural frequency and the damping factor to be easily determined:

$$\omega_n = \sqrt{\frac{K_d K_o}{N(\tau_1 + \tau_2)}} \tag{4.5}$$

$$\zeta = \frac{\omega_n}{2}\left(\tau_2 + \frac{N}{K_d K_o}\right) \tag{4.6}$$

The loop transfer function of Eq. (4.4) can now be rewritten in terms of ω_n and ζ:

$$H(s) = \frac{s\omega_n\left(2\zeta - \frac{\omega_n}{K_d K_o / N}\right) + \omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \tag{4.7}$$

The term K_dK_o/N is called the loop gain and if this is much greater than the natural frequency (which is nearly always the case) then the transfer function can be simplified to Eq. (4.8) [118]:

$$H(s) = \frac{2s\zeta\omega_n + \omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \tag{4.8}$$

A direct trade-off exists between the acquisition time (i.e. the time to lock to a signal) of the PLL and its output jitter. A small loop bandwidth will reduce the high frequency jitter, but slow the response. For applications such as demodulation, where the PLL output frequency will often change in response to input frequency changes, a larger loop bandwidth will be necessary. For applications such as clock recovery or frequency synthesis, where such a fast response is not necessary, a smaller loop bandwidth can be chosen which will reduce the jitter in the output signal. The choice of a 2nd order loop allows good settling response control, and loop stability to be easily achieved, whilst the choice of a lead lag filter allows the loop natural frequency and damping factor to be independently controlled.

4.2 Novel SI PLL architecture

The proposed novel 2nd order SI PLL has a similar architecture to the conventional PLL shown in Figure 4.1 but does not require a separate PD block. Due to the sampled nature of the SI technique, if the LF is clocked using the reference $\Theta'_2(s)$ signal it is possible to perform phase detection as part of the LF operation, which is referred to from this point on as 'implicit phase detection'. This so-called implicit PD is only suitable to take the place of a conventional PD due to what would normally be considered as non-ideal effects of the SI memory cells in the LF. It is for this reason that analysis is required to determine a suitable model for the implicit PD. Aside from the usual benefits of a SI implementation, the proposed SI PLL architecture has two additional advantages: firstly the use of a separate PD block can be avoided entirely, simplifying the loop design and secondly the overhead of having to generate an extra clock input for the LF is unnecessary. The three main loop blocks, the phase detector, loop filter and current controlled oscillator are now considered in detail.

4.2.1 Phase detector

In order to prove the concept of implicit phase detection and to model its behaviour for use in the proposed SI PLL, a large amount of mathematical analysis had to be carried out which is the subject of this section. Firstly, in Section 4.2.1.1 it is shown that if the SI LF is clocked by the $\Theta'_2(s)$ signal then phase detection is implicit, and a separate PD is not required. Secondly, in Section 4.2.1.2, a model for this implicit PD is developed so as to determine its suitability for use in the SI PLL architecture. Finally, in Section 4.2.1.3, the novel 2nd order SI PLL architecture is presented and practical guidelines for the design of the LF are developed, which allow the implicit PD model to be simplified to that of a conventional PD of constant gain, K_d.

4.2.1.1 Implicit phase detector concept

The principle of implicit phase detection is best illustrated through a voltage mode example of two cascaded sample and hold cells (switch and a capacitor) as shown in

Figure 4.3. The clocks ϕ_1 and ϕ_2 are non-overlapping, and the waveforms of the circuit are shown in Figure 4.4(a) and (b) for the case where V_{in} is a square wave voltage signal, with the same frequency as ϕ_1, but with a positive and negative phase error θ_e respectively. With a positive phase error the first sample and hold will *always* sample a high value since the input signal is *always* high when the sample switch opens. The second sample and hold cell will delay the first sample by a further half period, and the resulting output V_{out} will be a constantly high signal. For the case where there is a negative phase error, the opposite is true and the first sample and hold always samples a low value and so V_{out} is constantly low.

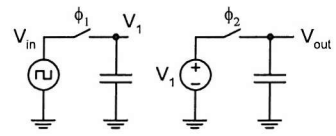


Figure 4.3 Voltage mode sample and hold

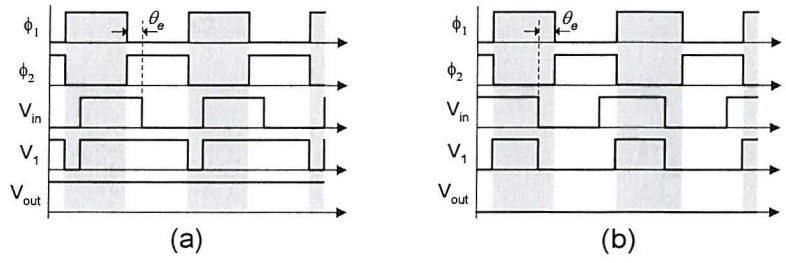


Figure 4.4 Voltage mode sample and hold waveforms with different phase errors

The circuit therefore behaves as a very crude PD, comparing the phase of ϕ_1 and V_{in} and giving an output, V_{out}, which can either be high or low. The phase detecting characteristics of the voltage mode sample and hold are shown in Figure 4.5 which defines the output for a given phase error θ_e. This PD has limited use in a PLL, which ideally requires a linear relationship (shown by the dashed line on Figure 4.5).

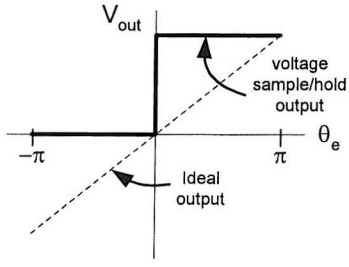


Figure 4.5 Transfer characteristic for voltage mode sample and hold

A SI memory cell is the current mode equivalent of Figure 4.3 and its operation as an implicit PD is now considered. Figure 4.6(a) shows a 2nd generation SI memory cell, which again requires two non-overlapping clocks, ϕ_1 and ϕ_2. During phase ϕ_1, M1 is diode connected and sinks both the bias, J and input, I_{in} current. During phase ϕ_2 the gate of M1 is disconnected and the memorized input current flows from the output I_{out}. This forms a current sample and hold circuit whose output is only available during ϕ_2. A small signal equivalent circuit is shown in Figure 4.6(b) where M1 is modelled with transconductance of g_m and gate source capacitance of C_{gs}. During ϕ_1 the circuit is simply a parallel RC network of C_{gs} and $1/g_m$ and the voltage V_{gs} increases or decreases depending on the direction of the input current.

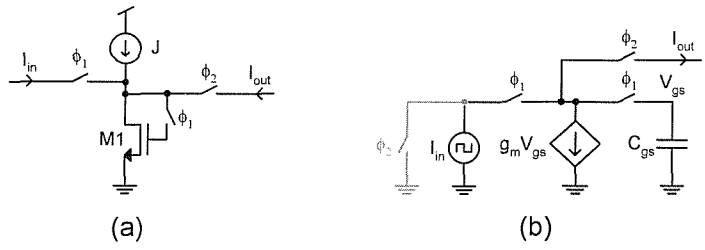


Figure 4.6 SI memory cell (a) and circuit model (b)

Figure 4.7(a) and (b) show circuit waveforms for Figure 4.6 with a square wave input signal of identical frequency to ϕ_1 and ϕ_2 but two different phase errors respectively. In Figure 4.7(a), during ϕ_1, V_{gs} first increases for a short period of time as the current flows into the capacitor, then decreases for a longer period as current flows out of the capacitor. A net negative current flows so V_{gs} is lower by the end of ϕ_1 than the start. During ϕ_2 V_{gs} is maintained at this value and the output current I_{out} flows. During the next ϕ_1 the net current is again just negative and so V_{gs} decreases further. I_{out} will settle to a constant value when V_{gs} is the same at the start of ϕ_1 as the end. In Figure 4.7(b) the input is high for longer than it is low during ϕ_1 and so V_{gs} and I_{out} rise until once more there is a zero net flow of charge onto the capacitor. Clearly the final, settled I_{out} samples are a function of the phase error between ϕ_1 and I_{in}, and a $\pi/2$ phase error would result in a near-zero output. The next section derives a model for the relationship between the settled I_{out} samples and the corresponding phase error, which can then be used to determine the design process for the proposed SI PLL.

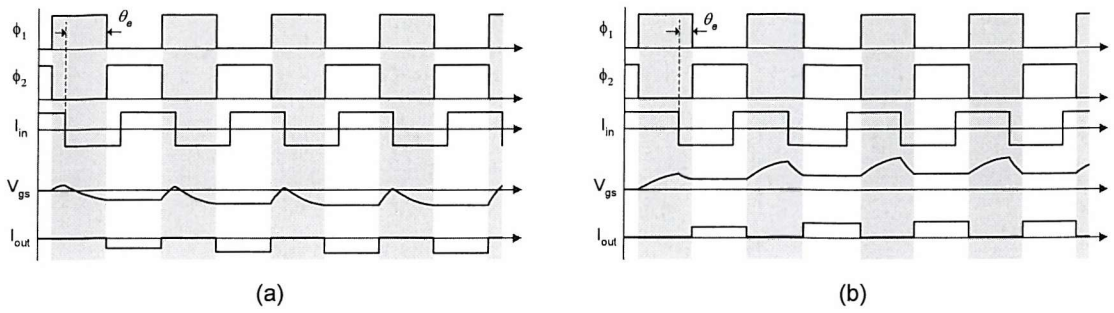


Figure 4.7 SI memory cell waveforms with different phase errors

4.2.1.2 Implicit phase detector model

A two part model is used to characterise the action of the implicit PD. The first part is an expression which relates the final, constant value of the I_{out} samples to the phase difference θ_e between ϕ_1 and I_{in} and furthermore gives a phase gain $K_d(\theta_e)$ for this characteristic. The second part is a single pole transfer function accounting for the time taken to settle to this final value. The first part of the model is now considered.

During phase one, the equivalent circuit of Figure 4.6(b) can be reduced to that of Figure 4.8, and to simplify the analysis, R and C have been used in place of $1/g_m$ and C_{gs}. The charge/discharge characteristics of this network and the case when the voltage is settled are shown in Figure 4.9(a) and (b) respectively. Only when the voltage at the start (before the charge) and at the end (after the discharge) of ϕ_1 are the same has the final value been reached. An expression must be derived which gives the final settled value of output current samples as a function of phase error, and this expression can then be differentiated to give $K_d(\theta_e)$. For purposes of the analysis t_1 is the period taken for the voltage to charge from V_1 to V_2, whilst current flows into the capacitor and t_2 is the period taken for the voltage to discharge from V_2 to V_1 whilst current flows out of the capacitor.

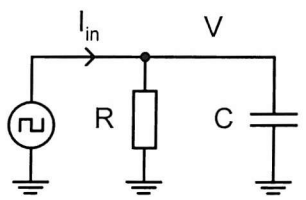


Figure 4.8 Equivalent input circuit during phase 1

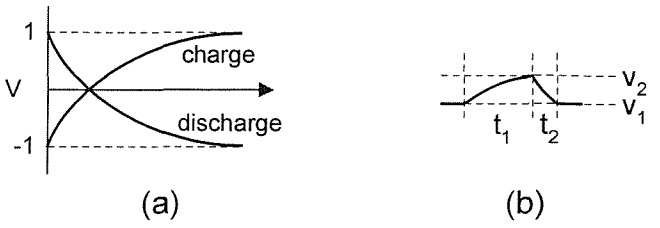


Figure 4.9 Charge and discharge characteristics (a) and the settled case (b)

The analysis first requires finding the voltage V_1 for a given t_1 and t_2. Assuming the maximum voltage obtainable is 1V and the minimum is –1V then the normal charge relations, assuming that the capacitor starts at either 1V or -1V, are:

$$\text{Charge:} \quad V_c(t) = -2\exp\left(\frac{-t}{RC}\right)+1 \tag{4.9}$$

$$\text{Discharge:} \quad V_d(t) = 2\exp\left(\frac{-t}{RC}\right)-1 \tag{4.10}$$

Manipulating Eq. (4.9) gives the complement t as a function of V_c:

$$t(V_c) = -RC\ln\left(\frac{V_c-1}{-2}\right) \tag{4.11}$$

The charging relation of Eq. (4.9) can be extended to include a time shift, T:

$$V_c(t) = -2\exp\left(\frac{-t-T}{RC}\right)+1 \tag{4.12}$$

Assuming a start voltage of V_s, Eq. (4.11) is substituted into Eq. (4.12) to obtain an expression giving voltage as a function of both start voltage and time:

$$V_c(t,V_s) = -2\exp\left(\frac{-t + RC\ln\left(\frac{1-V_s}{2}\right)}{RC}\right)+1 \qquad (4.13)$$

leading to: $$V_c(t,V_s) = -\exp\left(\frac{-t}{RC}\right)(1-V_s)+1 \qquad (4.14)$$

Similarly for the discharge:

$$V_d(t,V_s) = \exp\left(\frac{-t}{RC}\right)(1+V_s)-1 \qquad (4.15)$$

Substituting the times and voltages shown in Figure 4.9(b) into Eq. (4.14) and Eq. (4.15) gives:

V_1 to V_2 in time t_1: $$V_2 = -\exp\left(\frac{-t_1}{RC}\right)(1-V_1)+1 \qquad (4.16)$$

V_2 to V_1 in time t_2: $$V_1 = \exp\left(\frac{-t_2}{RC}\right)(1+V_2)-1 \qquad (4.17)$$

Solve for V_1:

$$V_1 = \exp\left(\frac{-t_2}{RC}\right)\left(\exp\left(\frac{-t_1}{RC}\right)(V_1-1)+2\right)-1 \qquad (4.18)$$

$$V_1\left(1-\exp\left(\frac{-t_2-t_1}{RC}\right)\right) = 2\exp\left(\frac{-t_2}{RC}\right)-\exp\left(\frac{-t_2-t_1}{RC}\right)-1 \qquad (4.19)$$

$$V_1 = \frac{2\exp\left(\frac{-t_2}{RC}\right) - \exp\left(\frac{-t_2 - t_1}{RC}\right) - 1}{1 - \exp\left(\frac{-t_2 - t_1}{RC}\right)} \qquad (4.20)$$

The result of Eq. (4.20) must now be used to derive an expression $I_{out}(\theta_e)$. Firstly, V_1 is in fact the gate voltage V_{gs} from Figure 4.6(b). Furthermore, assuming that the sampling period does not change significantly, then T_c (the sampling period at the PLL centre frequency) is considered fixed and relates t_1 and t_2 as follows:

$$\frac{T_c}{2} = t_1 + t_2 \qquad (4.21)$$

The times t_1 and t_2 can then be related to the phase error θ_e between the input frequency $\Theta_1(s)$ and the generated (and divided) ICO signal $\Theta'_2(s)$ which is used as the sampling frequency:

$$t_1 = \frac{\theta_e T_c}{2\pi} \qquad t_2 = \frac{T_c}{2} - \frac{\theta_e T_c}{2\pi} = \frac{T_c}{2}\left(1 - \frac{\theta_e}{\pi}\right) \qquad (4.22)$$

Substituting these values into Eq. (4.20) gives:

$$V_{gs}(\theta_e) = \frac{2\exp\left(\frac{-\frac{T_c}{2}\left(1-\frac{\theta_e}{\pi}\right)}{RC}\right) - \exp\left(\frac{-\frac{T_c}{2}\left(1-\frac{\theta_e}{\pi}\right) - \frac{\theta_e T_c}{2\pi}}{RC}\right) - 1}{1 - \exp\left(\frac{-\frac{T_c}{2}\left(1-\frac{\theta_e}{\pi}\right) - \frac{\theta_e T_c}{2\pi}}{RC}\right)} \qquad (4.23)$$

Simplifying Eq. (4.23), and using centre frequency F_c instead of period T_c, gives:

$$V_{gs}(\theta_e) = \frac{2\exp\left(\frac{\theta_e - \pi}{2\pi F_c RC}\right) - \exp\left(\frac{-\pi}{2\pi F_c RC}\right) - 1}{1 - \exp\left(\frac{-\pi}{2\pi F_c RC}\right)} \qquad (4.24)$$

Assuming an input current swing of between A and $-A$ and renaming C as C_{gs} and R as $1/g_m$ gives:

$$I_{out}(\theta_e) = A\left(\frac{2\exp\left(\frac{g_m(\theta_e - \pi)}{2\pi F_c C_{gs}}\right) - \exp\left(\frac{-g_m\pi}{2\pi F_c C_{gs}}\right) - 1}{1 - \exp\left(\frac{-g_m\pi}{2\pi F_c C_{gs}}\right)}\right) \qquad (4.25)$$

Eq. (4.25) is non-linear and Figure 4.10 shows the dependency of the linearity on the factor F_cC_{gs}/g_m. The relationship becomes more linear with large values of F_cC_{gs}/g_m.

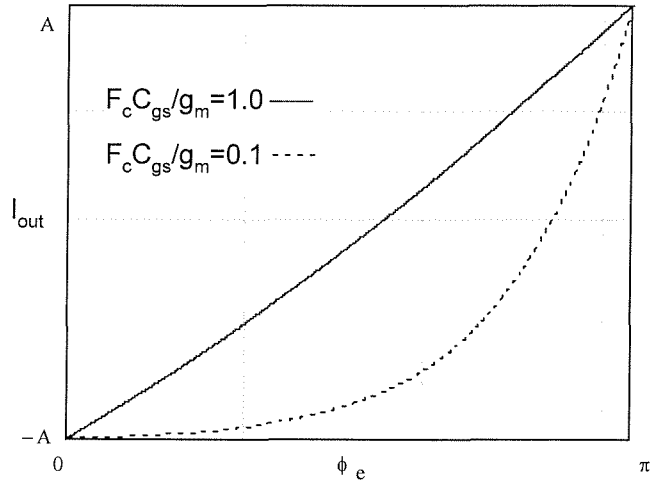


Figure 4.10 The first part of the implicit PD model

The first part of the implicit PD model is completed by obtaining the implicit PD gain K_d for any given phase error by differentiating the relationship of Eq. (4.25):

$$K_d(\theta_e) = \frac{g_m A \exp\left(\frac{-g_m(\pi-\theta_e)}{2\pi F_c C_{gs}}\right)}{\pi F_c C_{gs}\left(1-\exp\left(\frac{-g_m \pi}{2\pi F_c C_{gs}}\right)\right)} \quad (4.26)$$

The second part of the implicit PD model is now considered, and this should account for the time it takes for the current output samples to settle to the final value given by Eq. (4.25). A single pole transfer function dependent on C_{gs} and g_m is used to model this behaviour. Because only half the sample period is available to settle, the settling time constant is double the C_{gs}/g_m time constant giving the transfer function in Eq. (4.27). Figure 4.11 shows the completed two part model for the implicit PD.

$$H(s) = \frac{g_m}{g_m + 2sC_{gs}} \quad (4.27)$$

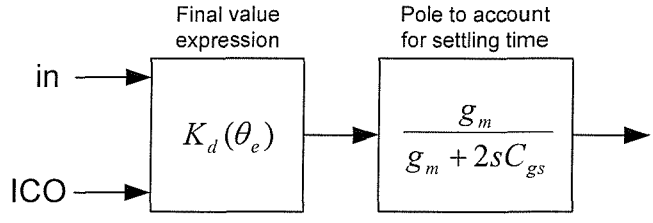


Figure 4.11 Complete two part implicit PD model

4.2.1.3 Practical implicit phase detector model

Having established the two part model for the implicit PD, the conventional PLL architecture of Figure 4.1 is modified to give the novel 2nd order SI PLL architecture shown in Figure 4.12. There is no need for a separate PD in the proposed SI architecture since the SI LF will achieve implicit phase detection due to the two part model given above. The loop transfer function for the proposed PLL of Figure 4.12 can be found:

$$H(s)=\frac{\frac{K_o}{N}\frac{1+s\tau_2}{(\tau_1+\tau_2)}\frac{g_m}{g_m+2sC_{gs}}K_d(\theta_e)}{s^2+s\frac{1+K_o\tau_2/N}{(\tau_1+\tau_2)}\frac{g_m}{g_m+2sC_{gs}}K_d(\theta_e)+\frac{K_o/N}{(\tau_1+\tau_2)}\frac{g_m}{g_m+2sC_{gs}}K_d(\theta_e)} \quad (4.28)$$

Clearly the transfer function of Eq. (4.28) is more complex than the simple 2nd order transfer function of Eq. (4.4) and it would therefore be a difficult task to design a PLL based on this. It is now shown that if certain guidelines are adhered to then the first part of the PD model (Eq. (4.26)) can be approximated to a constant gain K_d, and the second part (Eq. (4.27)) can be ignored, allowing the use of the standard 2nd order transfer function of Eq. (4.4).

First the pole introduced by the implicit PD is considered. If the pole frequency is significantly above the loop natural frequency ω_n then the system will remain stable, since this extra phase contribution will appear at a gain far less than unity. Existing literature concerning the placement of poles in higher order PLLs indicates that in practice loop stability is guaranteed if such an extra pole is placed at a frequency 10 times higher than ω_n [118].

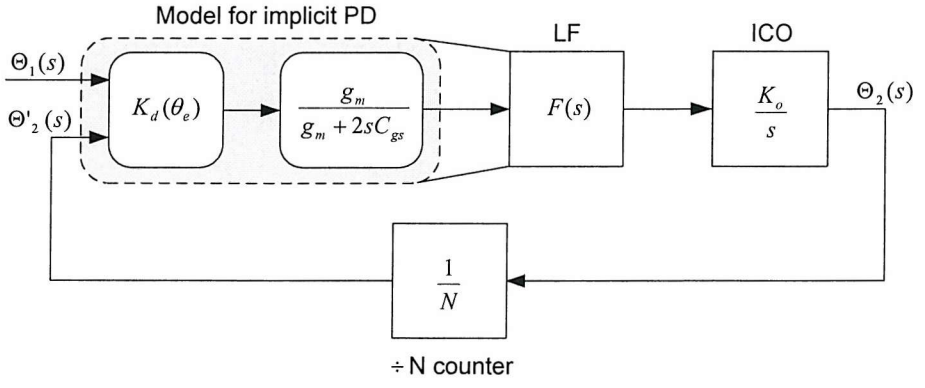


Figure 4.12 Novel 2nd order SI PLL architecture with implicit PD

In order to approximate $K_d(\theta_e)$, which is a function of phase error, to a constant K_d value it is necessary to determine the range of values $K_d(\theta_e)$ can assume and constrain this range. The maximum, $K_{d,max}$ and minimum, $K_{d,min}$ values of $K_d(\theta_e)$ occur when the phase error equals π and 0 respectively (see Figure 4.10):

$$K_{d,\min} = K_d(0) \qquad K_{d,\max} = K_d(\pi) \tag{4.29}$$

The ratio between these two values, $K_{d,rat}$, gives insight into the range of values of $K_d(\theta_e)$ and can be shown to equal:

$$K_{d,rat} = \frac{K_d(\pi)}{K_d(0)} = \exp\left(\frac{g_m}{2F_c C_{gs}}\right) \tag{4.30}$$

Eq. (4.30) is now used to determine the time constant (C_{gs}/g_m) necessary for a particular ratio, $K_{d,rat}$:

$$\frac{C_{gs}}{g_m} = \frac{1}{2F_c \ln(K_{d,rat})} \tag{4.31}$$

The value of the ratio, $K_{d,rat}$ has an impact on loop settling since it will determine how much the overall loop gain will vary over the acquisition period. This is familiar problem for designers of voltage-mode PLLs, since VCOs rarely have a very linear characteristic. A maximum allowable value for $K_{d,rat}$ of 2 has been chosen based on experience with these designs. From Eq. (4.31), the required time constant is:

$$\frac{C_{gs}}{g_m} = \frac{1}{2F_c \ln 2} \approx \frac{9}{12F_c} \tag{4.32}$$

Therefore if the SI memory cells used in the sampled SI LF are designed for the time constant given in Eq. (4.32) then this will ensure that the implicit PD will have a $K_{d,rat}$ of 2, and hence the assumption can be made that K_d is a fixed constant. In order to determine the feasibility of implementing this solution it is necessary to consider the transistor level design of the LF. Looking ahead to Section 4.3.3 it is found that unfortunately there is a contradictory requirement for the filter memory cell time constant in order to obtain an adequate settling accuracy:

(from Section 4.3.1) $$\frac{C_{gs}}{g_m} = \frac{1}{12F_c} \quad (4.33)$$

Satisfying Eq. (4.33) is essential for correct SI LF operation, however, in terms of the implicit PD this would correspond to a $K_{d,rat}$ of 400 which would no longer allow the assumption of a constant K_d. This implies that it is necessary to have two different time constants in place, one to satisfy the LF requirements and one for the implicit PD. Interestingly, the solution to this problem is very simple and requires no additional overheads. It is seen in Section 4.2.2 that the SI LF requires a SI sample and hold stage *before* the filter input to ensure that the filter output has no continuous time component. This sample and hold block is in a perfect location to 'tailor' to the requirements of the implicit PD. The filter can therefore be designed as usual with the filter memory cells satisfying the $1/12F_c$ time constant, and the SI sample hold stage can be altered to give the $9/12F_c$ time constant required to allow the constant K_d assumption. In order to determine the constant nominal K_d value to be used for the loop design, it is necessary to first find the phase error that corresponds to an output I_{out} of 0A by using Eq. (4.25):

$$\text{when } I_{out} = 0 \quad \theta_e = \left(\frac{2\pi F_c C_{gs}}{g_m}\right) \ln\left(\frac{1}{2} + \frac{1}{2}\exp\left(\frac{-g_m \pi}{2\pi F_c C_{gs}}\right)\right) + \pi \quad (4.34)$$

Now substituting Eq. (4.32) into Eq. (4.34) gives:

$$\theta_e = \frac{18\pi}{12} \ln\left(\frac{1}{2} + \frac{1}{2}\exp\left(-\frac{12}{18}\right)\right) + \pi = 1.8279 \text{ rad} \quad (4.35)$$

Using 1.8279 as θ_e in Eq. (4.26) gives the nominal value of $K_d(\theta_e)$:

$$K_d = 0.66A \text{ (Amps/rad)} \quad (4.36)$$

This constant K_d value is used in the rest of this chapter to design loop parameters for the novel 2nd order SI PLL architecture. Finally it is necessary to determine the constraint on ω_n in order to satisfy the condition that the pole introduced by the second part of the implicit PD model is at least 10 times greater in frequency. The frequency of the pole (in radians/sec) is given by Eq. (4.27) as $g_m/2C_{gs}$ and so the constraint is:

$$\frac{g_m}{2\omega_n C_{gs}} \geq 10 \qquad (4.37)$$

Substituting Eq. (4.32) into the above gives Eq. (4.38) which relates the ratio of the natural frequency of the loop (ω_n) to the PLL centre frequency ($2\pi F_c$). This shows that the loop natural frequency must be less than 1% of the PLL centre frequency in order for the pole introduced by the implicit PD to be ignored. From experience, the loop remains stable when this requirement is relaxed to 10%.

$$\frac{\omega_n}{2\pi F_c} \leq \frac{1}{30\pi} = 1.06\% \qquad (4.38)$$

This section has shown that in the context of a SI PLL, implicit phase detection can be achieved by the LF if it is clocked by the PLL output signal. A model has been derived for the implicit phase detector response (shown in Figure 4.11) which consists of the $K_d(\theta_e)$ relationship given in Eq. (4.26) and a single pole transfer function given in Eq. (4.27). This model was analysed further and it was found that it could be simplified to that of a conventional PD with a single K_d gain parameter if certain guidelines are followed. The guidelines include designing the memory cells in the LF sample and hold stage for a time constant 9 times longer than the memory cells in the rest of the filter, and ensuring that the loop natural frequency is less than 1% of the PLL centre frequency. Under these constraints, the implicit phase detector response can be modelled as a constant K_d gain given by Eq. (4.36).

4.2.2 Loop filter

In this section, the SI LF is designed to an ideal block level from the passive prototype filter structure, shown in Figure 4.2. The transfer function of the passive lead lag (single pole, single zero) filter is easily determined:

$$H(s) = \frac{1 + sCR_2}{1 + sC(R_1 + R_2)} \tag{4.39}$$

and if: $$\tau_1 = R_1 C \qquad \tau_2 = R_2 C \tag{4.40}$$

this gives: $$H(s) = \frac{H_o(s)}{H_i(s)} = \frac{1 + \tau_2 s}{1 + (\tau_1 + \tau_2)s} \tag{4.41}$$

Eq. (4.41) is divided through by s to give:

$$H_o(s) = \frac{1}{(\tau_1 + \tau_2)}\left[\tau_2 H_i(s) + \frac{1}{s}\left[H_i(s) - H_o(s)\right]\right] \tag{4.42}$$

The signal flow graph (SFG) can be drawn directly from Eq. (4.42) and is shown in Figure 4.13. It is clear from the SFG that the filter pole derives from the feedback path and the zero from the feed forward path.

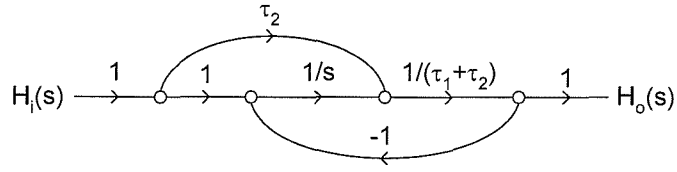


Figure 4.13 Signal flow graph of s-domain transfer function

Node quantities in the SFG can be scaled by a constant whilst keeping the transfer function the same by multiplying all branches going into the node by the constant and dividing all branches coming from the node by the same constant [14]. By

manipulating the SFG of Figure 4.13 in this way, the *1/s* factor can be changed to *2/sT*, ready for transformation into the z domain, as shown in Figure 4.14.

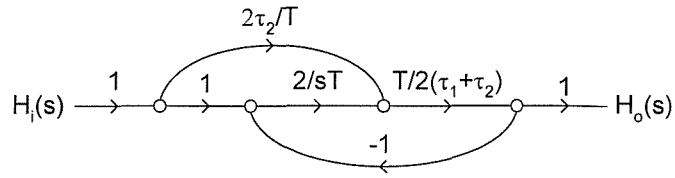


Figure 4.14 Modified s-domain signal flow graph

A sampled domain integrator is now required and the most popular choice is the bilinear integrator, which gives the best performance in SI circuits [14]. It is a simple matter to transform to the sampled domain by substituting *2/sT* for $1+z^{-1}/1-z^{-1}$ as shown in Figure 4.15. Equating branch values between Figure 4.14 and Figure 4.15 relates the coefficients α_1 and α_2 to τ_1, τ_2 and the sample period T:

$$\alpha_1 = \frac{2\tau_2}{T} \qquad \alpha_2 = \frac{T}{2(\tau_1 + \tau_2)} \tag{4.43}$$

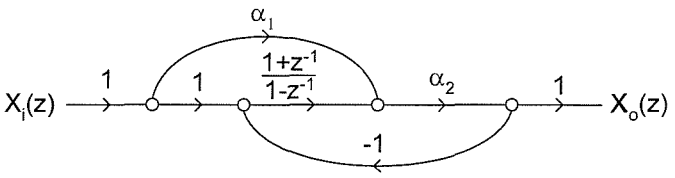


Figure 4.15 Signal flow graph for z-domain transfer function

In the case where the filter pole and zero frequencies are not much smaller than the PLL centre frequency, they should be pre-warped to account for the distortion when transformed into the sampled domain. For example to obtain a pre-warped value ω_p from the desired value ω, the following equation should be used:

$$\omega_p = \frac{2}{T}\tan\left(\frac{\omega T}{2}\right) \tag{4.44}$$

The equivalent block level circuit structure to the SFG in Figure 4.15 is shown in Figure 4.16.

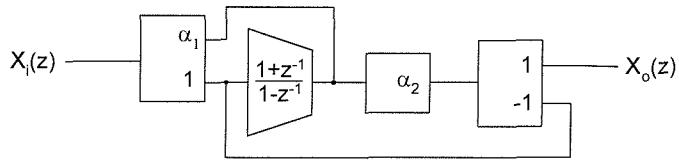


Figure 4.16 Equivalent circuit structure of the SFG

This structure can be simplified by altering the flow graph. Figure 4.17 shows how the last branch can be collapsed, reducing the implementation by one current mirror.

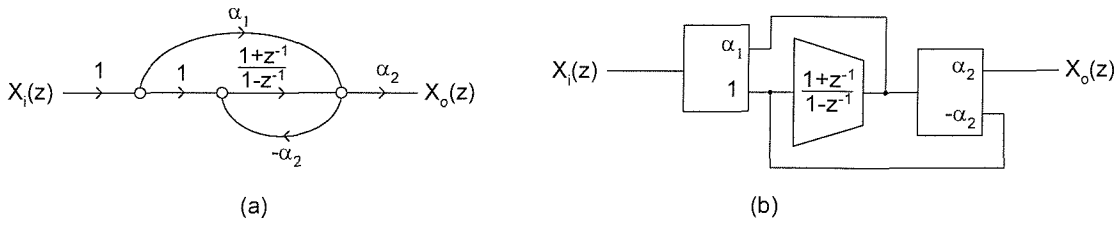


Figure 4.17 Reduced signal flow graph (a) and equivalent circuit structure (b)

An even simpler implementation is shown in Figure 4.18(a), where the elimination of a further current mirror is achieved through providing an extra copied output at the remaining two blocks. The coefficients can be redefined for clarity by renaming $\alpha_1\alpha_2$ with α_1, as shown in Figure 4.18(b) and these new coefficients can be calculated as follows:

$$\alpha_1 = \frac{\tau_2}{\tau_1 + \tau_2} \qquad \alpha_2 = \frac{T}{2(\tau_1 + \tau_2)} \tag{4.45}$$

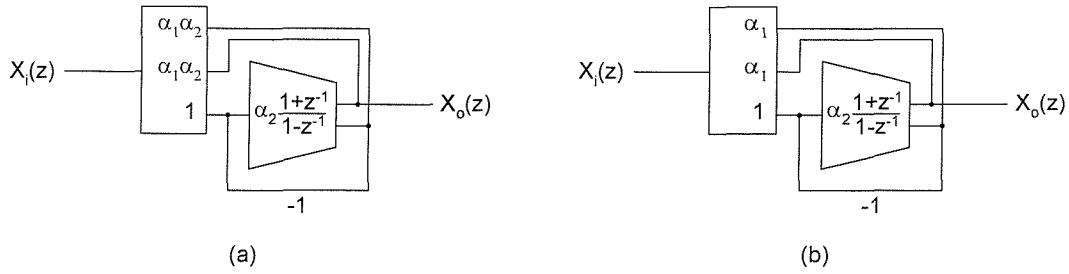


Figure 4.18 The further reduced structure (a) and renamed coefficients (b)

From Section 4.1.2, τ_1 and τ_2 are related to the PLL loop parameters as follows:

$$\tau_2 = \frac{2\zeta}{\omega_n} - \frac{N}{K_o K_d} \qquad \tau_1 = \frac{K_o K_d}{N\omega_n^2} - \tau_2 \qquad (4.46)$$

The SI LF coefficients of Figure 4.18(b) can therefore be calculated directly from the PLL loop parameters as follows:

$$\alpha_1 = \frac{N\omega_n}{K_o K_d}\left(2\zeta - \frac{N\omega_n}{K_o K_d}\right) \qquad \alpha_2 = \frac{N\omega_n^2 T}{2K_o K_d} \qquad (4.47)$$

This completes the ideal SI LF design. The loop damping factor and natural frequency can be adjusted independently of each other, which was the key reason for choosing a lead lag (single pole, single zero) filter. Because the filter contains a feed-forward path, a proportion of the filter input is coupled directly to its output signal which means that the output could contain a continuous time component as well as discrete samples. This is undesirable since the following ICO is not a sampled system so phase jitter is likely to increase. For this reason, it is essential to have a sample and hold stage before the filter input, and it is this sample and hold stage that is tailored to the implicit PD requirements as discussed in Section 4.2.1.3.

4.2.3 Current controlled oscillator

The most popular oscillator type used in PLLs is the ring oscillator, and this section briefly details the theory of their operation. The principle behind a three stage ring oscillator is shown in Figure 4.19 . Each stage produces a phase shift of $\pi/3$ causing a total phase shift of π (180°) and hence positive feedback and oscillation occurs.

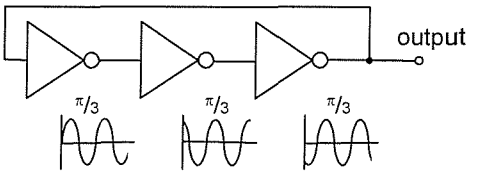


Figure 4.19 Three stage ring oscillator

The requirement for oscillation is that each stage produces a $\pi/3$ (60°) phase shift with at least unity gain at the natural frequency of oscillation, ω_o. If each stage has a dominant pole and a DC gain of A, these requirements are shown in Figure 4.20 where at the very point of oscillation, (when ω equals ω_o) the phase is $\pi/3$ and the magnitude response shows a gain of 0dB.

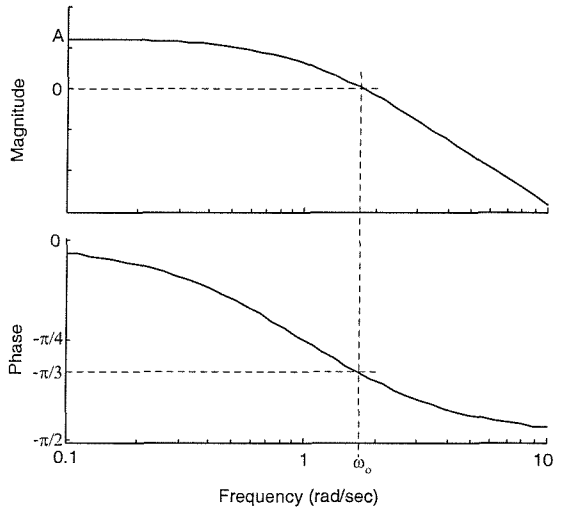


Figure 4.20 Condition for oscillation

Analysis is now given to determine the time constant and DC gain required for a particular frequency of operation. Assuming that a single stage approximates to a first order system of gain A and time constant τ, its transfer function is given by:

$$H(s) = \frac{A}{1+\tau s} \qquad (4.48)$$

where: $$|H(j\omega)| = \frac{A}{\sqrt{1+\tau^2\omega^2}} \qquad \angle H(j\omega) = -\tan^{-1}(\tau\omega) \qquad (4.49)$$

At the frequency of oscillation, ω_c, the phase is equal to $\pi/3$ giving:

$$\omega_c = \frac{1}{\tau}\tan\left(\frac{\pi}{3}\right) \qquad (4.50)$$

Also at the frequency of oscillation, in order for the oscillations to be sustained, the magnitude must be at least equal to unity, giving:

$$\frac{A}{\sqrt{1+\tan^2\left(\frac{\pi}{3}\right)}} \geq 1 \qquad (4.51)$$

Finally, from Eq. (4.50) and Eq. (4.51) the time constant required for a desired ω_c and the constraint on A for the system to oscillate can be determined:

$$\tau = \frac{1}{\omega_c}\tan\left(\frac{\pi}{3}\right) \qquad (4.52)$$

$$A \geq 2 \qquad (4.53)$$

This means that as long as the stages of a ring oscillator can be assumed to have a dominant pole, the oscillation condition can be met by designing for a gain of at least 2. Furthermore, the desired frequency of oscillation can be used to design the time constant of the single stages according to Eq. (4.52).

4.3 Transistor level design

The previous section detailed the design of the proposed novel 2nd order SI PLL architecture at a block level, introducing the high level equations necessary to link individual blocks to the PLL loop parameters. In this section, the detailed transistor level design is considered, building on the theory of the last section and leading to a complete schematic for the proposed SI PLL. The design of the class AB SI memory cell is of key importance to the overall PLL design and as such is considered first.

4.3.1 Class AB memory cell

Recently, an improved SI delay cell based on class AB operation has been proposed which can achieve reduced power consumption as compared to class A designs [25,

27, 28, 87-89]. Class A memory cells, such as the S^2I cell used in the filter design of Chapter 2, can be more accurate at reducing charge injection errors than their class AB counterparts, however, in the application of the proposed SI PLL the benefit of lower power motivates the choice of the newer class AB design. The class AB memory cell is shown in Figure 4.21 along with the clocking scheme necessary for its correct operation. The cell operates as follows: during ϕ_1 the cell is configured as two diode connected transistors, which source all the input and bias current present. Just before the end of ϕ_1, the phase ϕ'_1 ends, ensuring that the gate voltages of the NMOS and PMOS memory transistors are maintained and not disturbed when shortly afterwards the phase ϕ_1 ends and ϕ_2 simultaneously begins. The gate voltages on the memory transistors are maintained by their gate capacitances and so the same current continues to flow during ϕ_2, but this current now flows from the output through the ϕ_2 output switch.

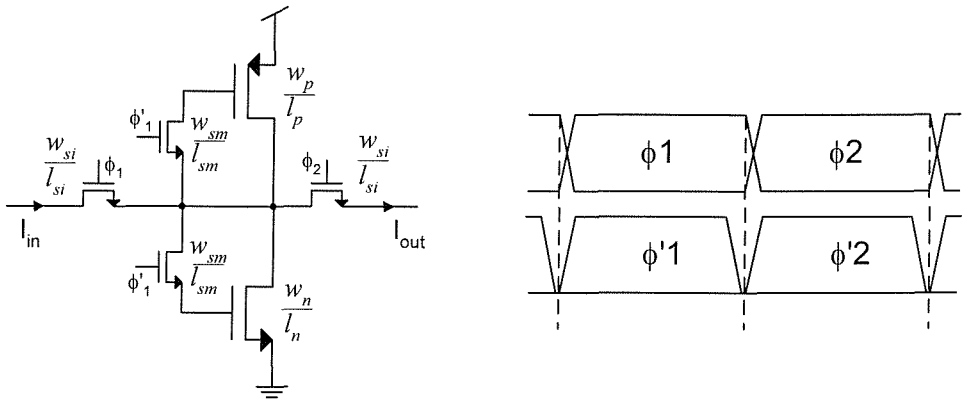


Figure 4.21 Class AB memory cell and clock waveform constraints

The bias current of a class AB memory cell is a function of the NMOS and PMOS transconductance (g_m) and the supply voltage (V_{dda}). This dependence on supply voltage has been addressed in the literature through the use of documented compensation mechanisms (which are not discussed here) [87]. Because the cell has only two transistors between the supply rails, it is possible for it to function at very low supply voltages, which further reduces power consumption. However, if these low voltages are also used for the clock signals to the NMOS switches it is possible that during large input signal swings the V_{gs} of the NMOS switches may reduce to the extent that they conduct when off. This problem can be circumvented by driving

the switch gates with levels higher than the V_{dda} supply voltage. This has motivated the use of two different voltage supplies in the proposed SI PLL, a lower voltage supply (V_{dda}) for the analogue memory cells, and a higher voltage supply (V_{dd}) for the digital clock structures which provide the control signals for the switches.

The design of the class AB memory cell is now detailed, using an example specification of 10MHz sampling frequency and 100μA maximum input signal. First, the bias current for the cell is determined, and it can be shown that a bias current of ¼ the maximum signal swing is necessary for correct operation [87]:

$$J = \frac{I_{\max}}{4} \qquad (4.54)$$

where J is the bias current, and I_{max} is the maximum expected signal amplitude. With an I_{max} of 100μA a bias current of 25μA can be used. In order to maintain saturated operation the gate overdrive voltages must be set to half a threshold voltage [87] allowing the supply voltage to be chosen to give the correct headroom:

$$V_{dda} = 2(V_{gt} + V_t) = 3V_t \qquad (4.55)$$

where V_{dda} is the analogue (memory cell) supply voltage, V_{gt} is the gate overdrive voltage and V_t is the threshold voltage. The threshold voltage for the process used in this example is 0.53V, which makes V_{gt} 0.265V and this gives a (analogue) supply voltage V_{dda} of 1.6V. The transconductance, g_m of the cell required for a given bias current and gate overdrive voltage is given by:

$$g_m = \frac{J}{V_{gt}} \qquad (4.56)$$

In this example g_m is therefore 94.33μS. Six time constants are allowed for settling which gives good accuracy (equivalent to better than 8 bits) and with a settling period of half the sampling frequency the gate capacitance can be calculated:

$$C_{gs} = \frac{g_m}{12F_s} \tag{4.57}$$

where C_{gs} is the gate capacitance and F_s is the sampling frequency and it is assumed that the gate capacitance is the dominant capacitance at the common node. In this example C_{gs} is calculated as 786.2fF. The width and length of the NMOS and PMOS transistors, w_p/l_p and w_n/l_n are given by [14]:

$$w_p = \sqrt{\frac{g_m C_{gs}}{\beta_p V_{gt} C_{gu}}} \qquad l_p = \frac{C_{gs}}{C_{gu} w_p} \tag{4.58}$$

$$w_n = \sqrt{\frac{g_m C_{gs}}{\beta_n V_{gt} C_{gu}}} \qquad l_n = \frac{C_{gs}}{C_{gu} w_n} \tag{4.59}$$

where β_p and β_n are the PMOS and NMOS transconductance parameters respectively and C_{gu} is the capacitance per unit area of the gate region. The process used in this example has values for β_p as 18μA/V^2, β_n as 55μA/V^2 and C_{gu} as 3.1fF/μm^2 which gives initial PMOS and NMOS dimensions of 70.80/3.60 and 40.50/6.25 rounded to the technology grid (0.05μm). All that remains is to find values for the memory and input/output switch dimensions w_{sm}/l_{sm} and w_{si}/l_{si}. In the interest of charge injection, for both switches the length is set to the minimum length allowed by the technology, 0.35μm. For the memory switch the width needs to be designed such that the 2nd order loop formed by the gate capacitance, C_{gs}, the drain capacitance, C_d, and the switch conductance, g_s settles quickly. The drain capacitance which was neglected in the memory transistor calculations is given by:

$$C_d = C_{du} L \tag{4.60}$$

where C_{du} is the capacitance per unit length of drain region, a process related parameter which in this example is 1fF/μm. This gives C_d as 3.6fF for the PMOS and

6.25fF for the NMOS (mean of 4.925fF). Note that it is clear that this contribution is indeed small compared to the gate source capacitance (786.2fF) and as such it was reasonable to assume that the gate source capacitance was the dominant capacitance at the common node. The memory switch width, w_{sm}, is given by Eq. (4.61) [87].

$$w_{sm} = \frac{4l_{sm}g_m(C_s + C_d)}{g_{su}C_{gs}} \quad (4.61)$$

where g_{su} is the unit switch on conductance, and C_s is an estimate of the switch stray capacitances and interconnect capacitances, which in total are estimated to be $0.2C_{gs}$ [14]. In this example, g_{su} is 60μS, and this gives a memory switch width of 0.45μm rounded to the technology grid. In order to calculate the input/output switch widths, a heuristic is derived based on an engineering judgement in [14], which specifies that the switch should not drop more than 10% of the gate overdrive voltage with an input current equal to the memory cell bias current:

$$w_{si} = \frac{10Jl_{si}}{V_{gt}g_{su}} \quad (4.62)$$

This gives a width of 5.50μm, rounded to the grid. The entire first cut design is now complete and simulations can be used to improve on this set of dimensions, which are only as accurate as the square law models from which they are derived. The final set of dimensions is shown in Table 4.1.

Table 4.1 Class AB memory cell dimensions

Transistor	Dimensions (μm)
PMOS memory w_p/l_p	76.05/3.10
NMOS memory w_n/l_n	40.50/5.15
Input/Output switch w_{si}/l_{si}	5.50/0.35
Memory switch w_{sm}/l_{sm}	0.70/0.35

The accuracy of the class AB memory cell can be greatly improved by using a differential structure. This is not just because a differential structure rejects common

mode signals such as noise but also because when both complementary signals are available, neutralisation capacitors can be used to dramatically reduce the cell's effective output conductance [87]. Figure 4.22 shows a differential class AB memory cell, including all switches, dummy switches (for charge injection improvements), and neutralisation capacitors:

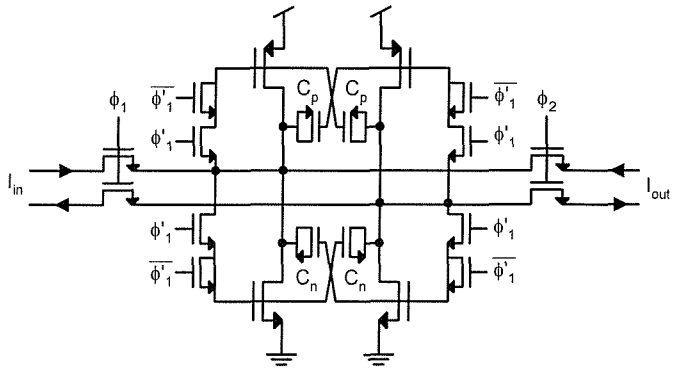


Figure 4.22 Differential, neutralised class AB memory cell

During ϕ_1 the input current is split between two identical class AB cells. During ϕ_2, both cells contribute their memorised current to the output. The neutralisation capacitors, C_p and C_n, greatly improve the cell's output conductance. In normal cells, voltage disturbances at the drains are coupled to the memory transistors gates via their parasitic drain gate overlap capacitance and this produces a loss error in the output signal. By placing capacitors from the gate to the drain of the opposite cell, the voltage disturbances are coupled with inversion to the memory transistor's gate and hence any residual errors are neutralised [87].

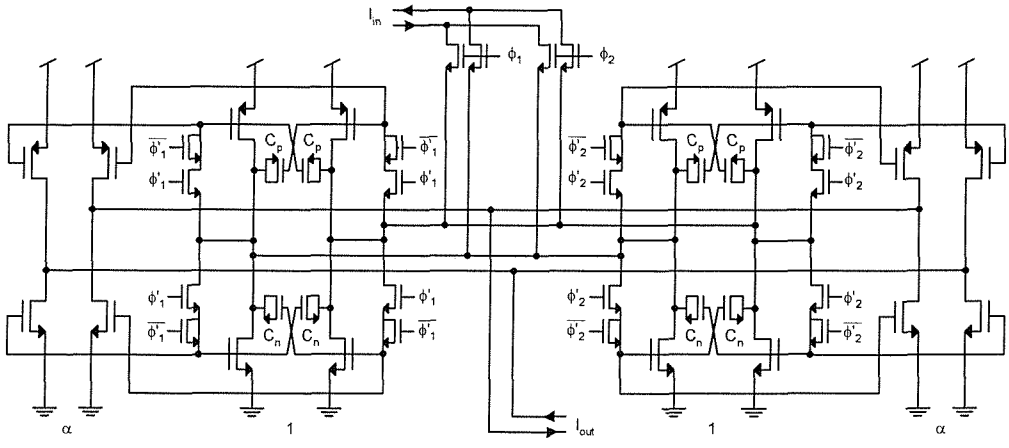


Figure 4.23 Double sampled, differential class AB SI bilinear integrator

The SI filter designed in Section 4.2.2 requires a bilinear integrator. A high performance double sampled bilinear integrator can be found in the literature, and is shown in Figure 4.23 [87]. The circuit contains two cross coupled differential delay cells, one operating on ϕ_1 and one on ϕ_2, with input switches routing the input to either side in turn. The output consists of the summation of the present input sample, the last input sample and the last output sample and is scaled by the constant α. Figure 4.24 is a full transistor level transient response showing the differential output of the bilinear integrator with a DC input, using dimensions given in Table 4.1. The cell clearly has sufficient time for settling at the designed sampling frequency of 10MHz. The accuracy and linearity is also excellent and this can be mainly put down to the use of the neutralisation technique [87].

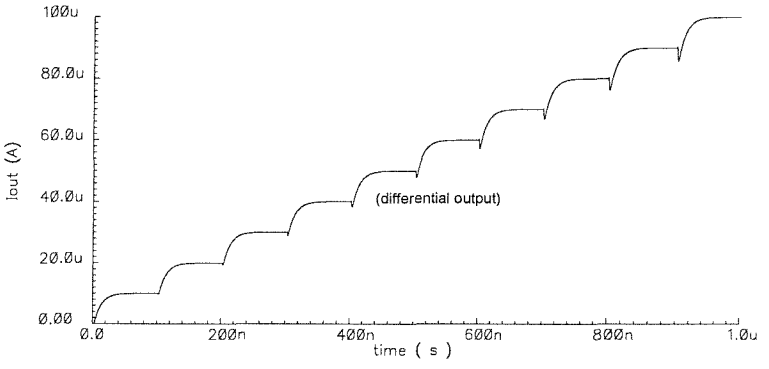


Figure 4.24 Output of the double sampled bilinear integrator with a DC input

This completes the design of the class AB memory cell building blocks which are now used to implement the phase detector (sample and hold cell) and loop filter.

4.3.2 Phase detector

The theory outlined in Section 4.2.1 covering the implicit PD operation was based on a class A single sampled single ended circuit. In order to achieve compatibility with the double sampled, fully differential filter which will be designed from the memory cells detailed in Section 4.3.1 the implications of double sampling on the model for the implicit PD must be considered. A double sampled SI sample and hold stage implemented using the class AB memory cell detailed in the previous section is shown in Figure 4.25.

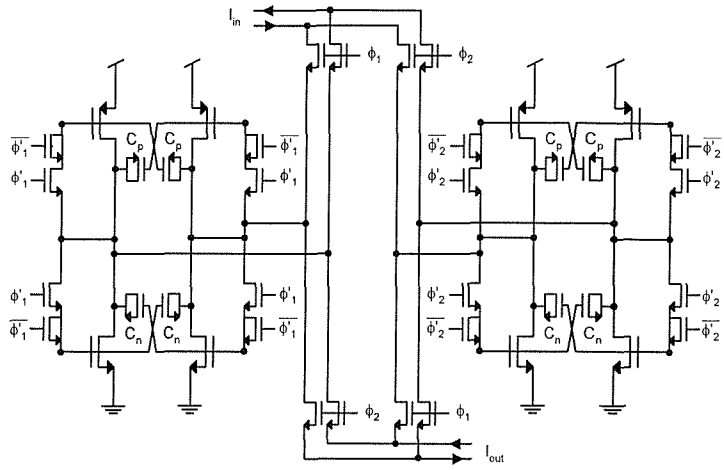


Figure 4.25 Double sampled class AB differential sample and hold stage

The input switches direct the differential input current to the left or right side of the cell during ϕ_1 and ϕ_2 respectively. The output switches allow the stored current to flow from the left or right side of the cell during phases ϕ_2 and ϕ_1 respectively, thus delivering a stored sample twice during the complete clock period, and leading to the double sampled nature of the circuit. In fact the cell shown in Figure 4.25 has been modified for use as the PD circuit, and this modification is now discussed.

The model presented in Section 4.2.1 relied on the charge/discharge cycle time being equal to the sampling phase of the memory cell, which in the case of the single-sampling model used was half the sampling clock period. Normally, double sampled circuits require a clock of only half the sample clock frequency of an equivalent single sampled circuit, because the sampling occurs twice during the sample clock period. However, in the context of the implicit PD, halving the sample clock frequency would result in a net zero output, rendering the implicit PD useless. The reason for this is shown in Figure 4.26 (refer to Figure 4.7), where it is clear that although half the clock frequency still gives the same number of samples per unit time, doubling the actual sampling period means that the net flow of current into the memory cell during one of those sample periods is always zero.

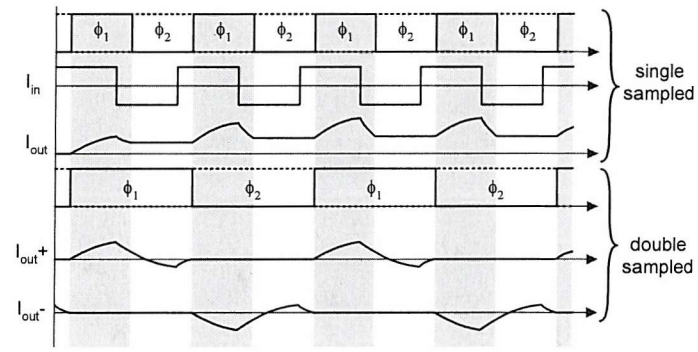


Figure 4.26 Double sampling with half the clock used for single sampling

So, despite using a double sampling cell, the sampling frequency must be the same as in the single sampled case, as shown in Figure 4.27. The output now gives the correct magnitude, identical to the single sampled case but provided during the entire clock period due to the double sampling. One problem still exists however: in the double sampled case, the output will oscillate between a positive and a negative version of the single sampled output, because the input appears inverted during ϕ_2. This can easily be addressed, by inverting the ϕ_2 output samples, and at circuit level, due to the use of differential signals this is achieved by swapping over the output signal paths of the ϕ_2 (right) side of the sample and hold cell. This is the only structural modification necessary to use the double sampled sample and hold cell for the implicit PD model, and this modification has been implemented in Figure 4.25.

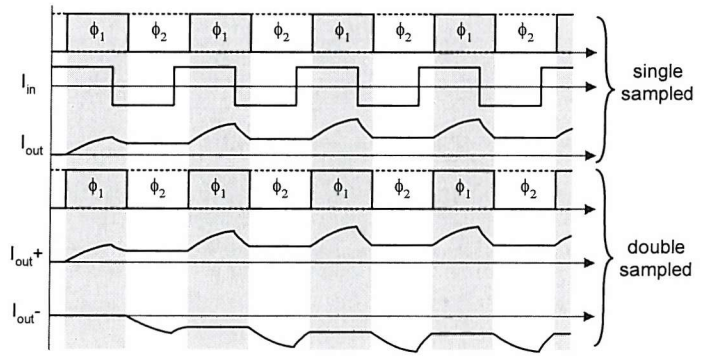


Figure 4.27 Double sampling with the same clock used for single sampling

The memory cells of the sample/ hold stage in the SI LF need to be designed for a time constant of 9 times longer than those used in the rest of the LF, in order for the implicit PD to be modelled by a constant K_d value (see Section 4.2.1). This requires

making the area of the sample and hold memory cell transistors 9 times greater than those in rest of the LF. This is trivial to implement and simply requires the widths and lengths to be scaled by 3 for the sample and hold memory cell transistors:

$$w_{p,pd} = 3w_p \qquad l_{p,pd} = 3l_p$$
$$w_{n,pd} = 3w_n \qquad l_{n,pd} = 3l_n \tag{4.63}$$

where $w_{p,pd}$, $w_{n,pd}$, $l_{p,pd}$ and $l_{n,pd}$ are the altered memory transistor dimensions used in the sample and hold cell and w_p, w_n, l_p and l_n are the memory cell dimensions used in the rest of the SI LF. Using the dimensions from Table 4.1 (but with memory transistor widths and lengths scaled by 3) it is possible to derive a graph of the simulated transistor level implicit PD transfer function, similar to Figure 4.10. This is achieved by simulating the cell with an input square wave current with frequency equal to the sample clocking waveform but delayed by some factor. Once the steady state has been reached, this value should represent the $I_{out}(\theta_e)$ relationship of Eq. (4.25), and by running a number of simulations with different delay values, a picture can be built up of the phase transfer function. This phase transfer function is shown in Figure 4.28. At zero output, the simulated PD gain (i.e. the slope of Figure 4.28) is found as 6.2μA/rad which is very close to the value predicted by Eq. (4.36) of 6.6μA/rad.

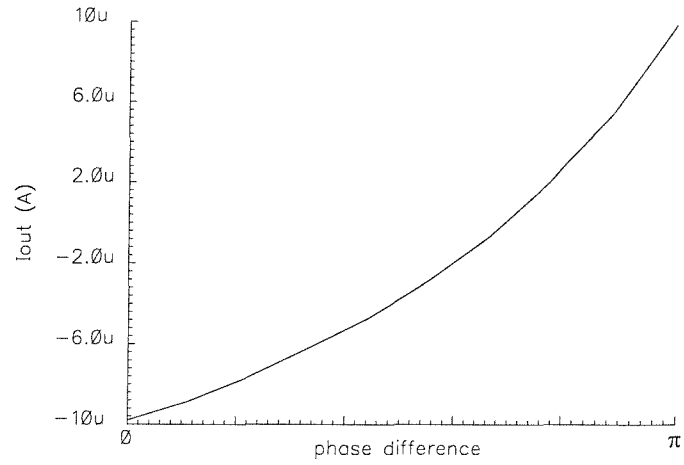


Figure 4.28 Simulated implicit PD transfer function

4.3.3 Loop Filter

The LF theory outlined in Section 4.2.2 coupled with the memory cell design in Section 4.3.1 is sufficient to produce the filter transistor level design. The complete transistor level filter structure is shown in Figure 4.29 where it can be seen that simple class AB current mirrors are used for the input mirror block of the filter, and that the bilinear integrator cell of Figure 4.23 has been extended to provide an extra differential output. Figure 4.29 excludes the sample and hold block for clarity, since this is already shown in Figure 4.25.

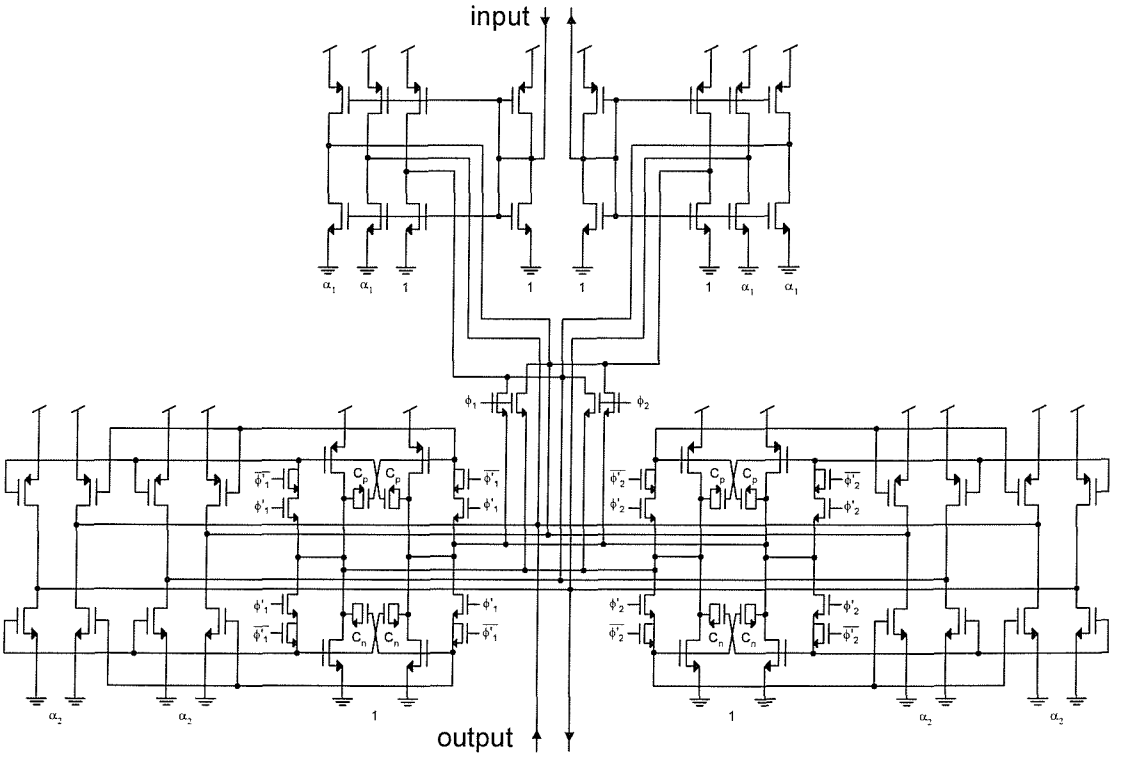


Figure 4.29 Transistor level SI loop filter

4.3.4 Current controlled oscillator

Most current controlled oscillators (ICOs) used in conventional PLL designs are based on a ring oscillator [120-126] governed by the theory outlined in Section 4.2.3. The approach used for the proposed SI PLL is based on the design in [120, 123, 126] where each stage of the ring is a simple current steering amplifier (CSA) cell (Figure 4.30) which consists of a current source, I_b, and a pair of NMOS devices. M1 is the

input device and M2 is the load. When V_{in} is high M1 turns on sinking the bias current I_b while M2 shuts off, and under this condition the on resistance of M1 defines the output low voltage. When V_{in} is low M1 turns off and I_b is steered to M2, and under this condition the resistance of the diode connected M2 defines the output high voltage. With three CSA stages and by varying I_b, a current controlled CSA-based ring oscillator is formed with an output swing given by Eq. (4.64) [120].

$$\Delta V = V_{th} + \sqrt{\frac{(W/L)_1 - (W/L)_2}{(W/L)_1 (W/L)_2} \cdot \frac{2I_b}{\beta_n}} \qquad (4.64)$$

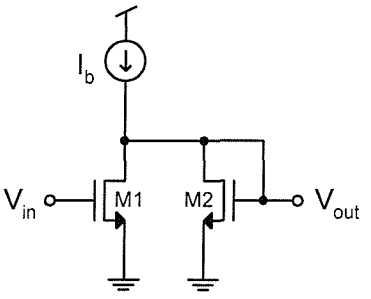


Figure 4.30 Current steering amplifier for use in the ICO

The voltage difference varies with $\sqrt{I_b}$ and hence the voltage swing of the cell increases with frequency. This is a desirable feature since the signal level improves at high frequency when normal circuit noise is likely to increase. The current source buffers the output from positive supply variations and is in practice provided by a current mirror. The total bias current comprises a DC bias, giving the centre frequency and an additional signal current which gives the frequency change. Since this signal current is in differential form from the SI LF output, a differential to single ended converting circuit is required. Furthermore, the ICO is powered from the lower analogue supply voltage, V_{dda} in order to save power, yet since the clock logic operates off the higher V_{dd} supply, a level shifting circuit is required to shift and amplify the low voltage oscillations to the correct levels. The total ICO scheme employed in this work is shown in Figure 4.31. The current mirrors used for the input converter stage and the oscillator current source are all dimensioned the same as the class AB memory cell, so that the quiescent points are well matched. The CSA

transistors are designed such that the total capacitance at the output node gives the desired oscillation frequency (i.e. the centre frequency). The amplifier is a simple active load amplifier, with separate bias current I_{bias}. Because the amplitude of the oscillator output varies with frequency, the buffer output has a varying duty cycle which is undesirable for correct SI memory cell operation. To solve this problem, the oscillator is designed for twice the required frequency and the buffer output is divided by two to restore a 50% duty cycle. The gain of the ICO can be altered once it has been designed by adding a current mirror scaling stage between the converter and oscillator block.

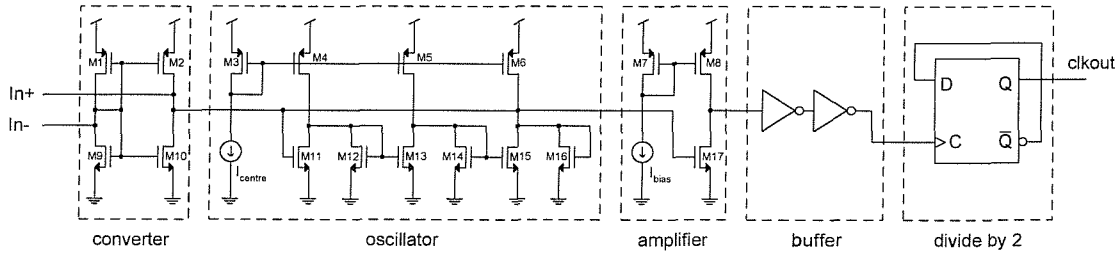


Figure 4.31 Current controlled oscillator

The frequency of operation can be determined analytically, through the use of the theory outlined in Section 4.2.3 and using the small signal equivalent circuit of a single CSA, shown in Figure 4.32. The sum of capacitance at the common node is modelled as C_{tot} and g_{m1} & g_{m2} are the transconductances of M1 and M2 respectively.

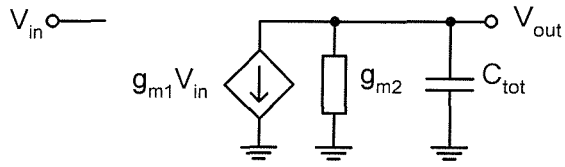


Figure 4.32 Small signal equivalent circuit of the CSA

The voltage gain of each stage is given by:

$$\frac{V_{out}}{V_{in}} = -\frac{g_{m1}}{g_{m2}} \qquad (4.65)$$

This ratio is constrained according to the analysis in Section 4.2.3 by Eq. (4.66). In practice, in order to ensure strong oscillation over the entire frequency range, g_{m1} is made about five times g_{m2} as shown by Eq. (4.67).

$$\frac{g_{m1}}{g_{m2}} \geq 2 \qquad g_{m1} \geq 2g_{m2} \tag{4.66}$$

$$g_{m1} = 5g_{m2} \tag{4.67}$$

The time constant of the small signal circuit in Figure 4.32 is given by:

$$\tau = \frac{C_{tot}}{g_{m2}} \tag{4.68}$$

From Eq. (4.50) this time constant relates to the oscillation frequency as follows:

$$\tau = \frac{C_{tot}}{g_{m2}} = \frac{1}{\omega_c}\tan\left(\frac{\pi}{3}\right) \tag{4.69}$$

The total capacitance is given by the sum of the M1 and M2 gate and overlap capacitances:

$$C_{tot} = C_{gu}\left(W_1L_1 + W_2L_2\right) + C_{du}\left(L_1 + L_2\right) \tag{4.70}$$

In the steady state the fraction of bias current that flows in g_{m2} is given by g_{m2}/ g_{m1} and so g_{m2} is given as follows:

$$g_{m2} = \frac{2I_D}{V_{gt}} = \frac{2I_b}{5V_{gt}} \tag{4.71}$$

where V_{gt} is the gate overdrive voltage of M2. From Eq. (4.69) and Eq. (4.71) the bias current is found to be directly proportional to the frequency of oscillation, as shown in Eq. (4.72) and the frequency gain (K_o) is given in Eq. (4.73). Using these equations and simulation, suitable dimensions for an ICO with a 20MHz centre frequency are given in Table 4.2. The architecture allows fine adjustment of the centre frequency to be made after implementation by altering I_{centre}.

$$I_b = \frac{5\omega_c V_{gt} C_{tot}}{2\tan\left(\frac{\pi}{3}\right)} \qquad (4.72)$$

$$K_o = \frac{5V_{gt} C_{tot}}{2\tan\left(\frac{\pi}{3}\right)} \qquad (4.73)$$

Table 4.2 ICO transistor dimensions

Transistor	Dimensions (μm)
I_{centre}	26μA
I_{bias}	15μA
M1-M6	76.05/3.1
M9, M10, M17	40.50/5.15
M11, M13, M15	38.5/4
M12, M14, M16	2/1
M7, M8	50/0.35
M17	30/0.35

A transient response with zero input current is shown in Figure 4.33. The oscillator output amplitude is relatively small at around 1V pk-pk and the amplifier increases this value such that it crosses the threshold points of the two inverters acting as a buffer for the signal. Finally the output of the divide by 2 flip flop gives the desired frequency at a 50% duty cycle. The plot of frequency versus input current for the complete block is shown in Figure 4.34. The frequency gain is reasonably linear over a large range (3-15MHz), varying from 0.45 to 0.75MHz/μA. It is likely that for a relatively low frequency design such as this, improved linearity may be possible

through the use of more stages in the oscillator, or by designing the oscillator for a higher frequency and dividing the output further.

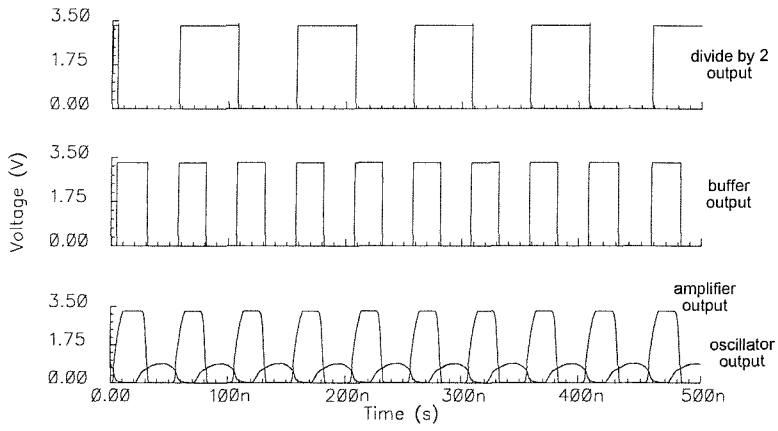


Figure 4.33 Time domain ICO performance

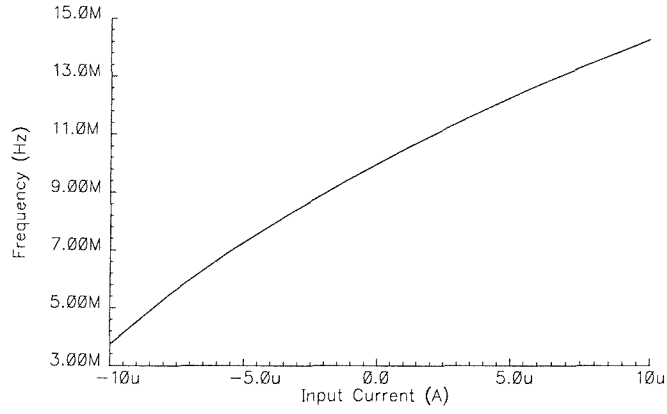


Figure 4.34 Frequency domain ICO performance

4.3.5 Clock generation and input circuitry

For correct operation of the class AB memory cells in the implicit PD (sample and hold) and filter block, it is essential to generate clock phases to the specification of Figure 4.21. A delay line of inverters is employed to obtain slightly shifted versions of the input clock and output logic converts these signals into the correct phases, as shown in Figure 4.35. Extra capacitors are used to load the intermediate inverter outputs in order to obtain exact timing relationships [13]. Clock buffers are used to drive the clock lines and correct operation is confirmed by the transient simulation shown in Figure 4.36 for an input clock frequency of 200MHz.

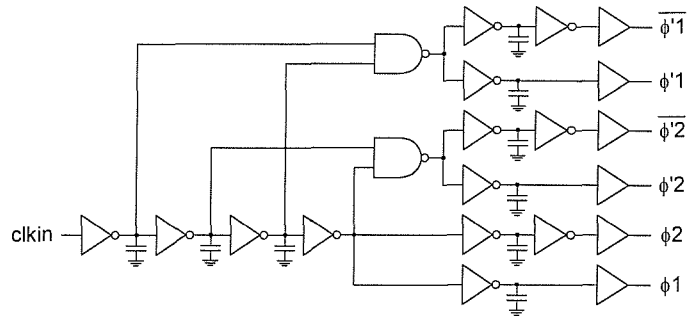


Figure 4.35 Clock generation scheme for class AB memory cell

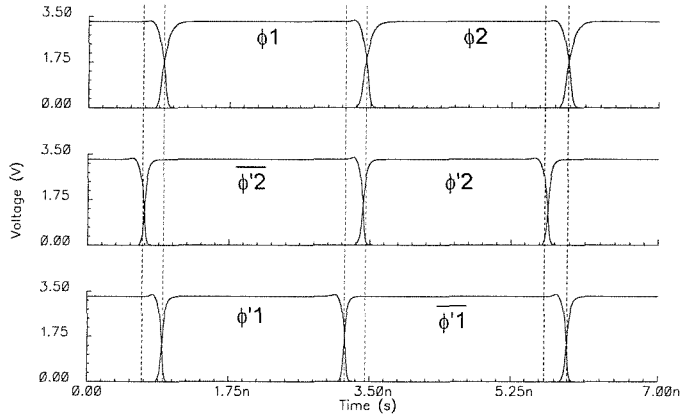


Figure 4.36 Time domain simulation confirming clock generator operation

The input to the proposed SI PLL is a square wave voltage signal but for correct operation, the input to the SI LF sample and hold must be a square wave *current* signal of amplitude *A*. Due to the nature of the differential design the solution can be achieved by using a fixed DC current source and switches at the input to the PLL to swap the input signal paths during half of the input clock phase, as shown in Figure 4.37.

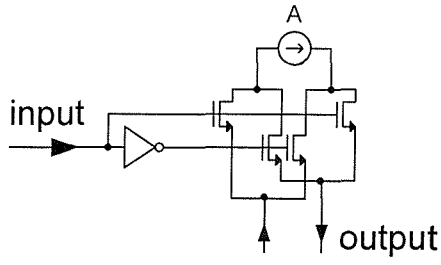


Figure 4.37 Input circuitry

4.4 AutoPLL

The SI technique facilitates analogue functionality on a standard digital CMOS process and at low supply voltages, allowing the integration of analogue cores on present and future System on Chips (SoCs). Despite these excellent attributes and prospects, the technique is a relatively unfamiliar design approach and as such, systematic design methods and tools are essential for its widespread development and use. Chapter 3, Section 3.2 presented a generic CAD methodology suitable for SI analogue cores and this is now used to automate the novel 2^{nd} order SI PLL design flow detailed in this chapter in Sections 4.2 and 4.3. This SI PLL methodology, named AutoPLL, consists of both a cell hierarchy and CAD tool to allow the rapid design of SI PLLs to transistor level. The AutoPLL cell library and tool are implemented in an industry standard EDA suite, Cadence Design Framework II (DFII), allowing their seamless use during a normal design flow. Sections 4.4.1 and 4.4.2 present the AutoPLL cell library and tool respectively.

4.4.1 AutoPLL cell library

The AutoPLL cell hierarchy, illustrated by Figure 4.38, has been developed to support the design of the SI PLL architecture proposed in this chapter. The hierarchy extends from the top level PLL core symbol to the bottom level CMOS BSIM3v3 foundry models, with three intermediate levels of abstraction in between. The level below the top level PLL is a structural block level (level 4) containing the major PLL circuit blocks such as the LF, PD and memory cell clock phase generator. Level 3 breaks these major blocks into smaller sub-blocks, for example a class AB current mirror and bilinear integrator. Behavioural models created from ideal macro cells or written in VerilogA could be used at this stage to enable fast simulations and optimisation of the PLL parameters. At the next level down (level 2) lie the transistor level schematics for the level 3 sub-blocks. All transistor schematics use generic three terminal MOS transistor symbols to hide the four terminal process vendor MOS models. Technology retargeting can be easily achieved by swapping these lowest level foundry transistor instances for the models from an alternative process library.

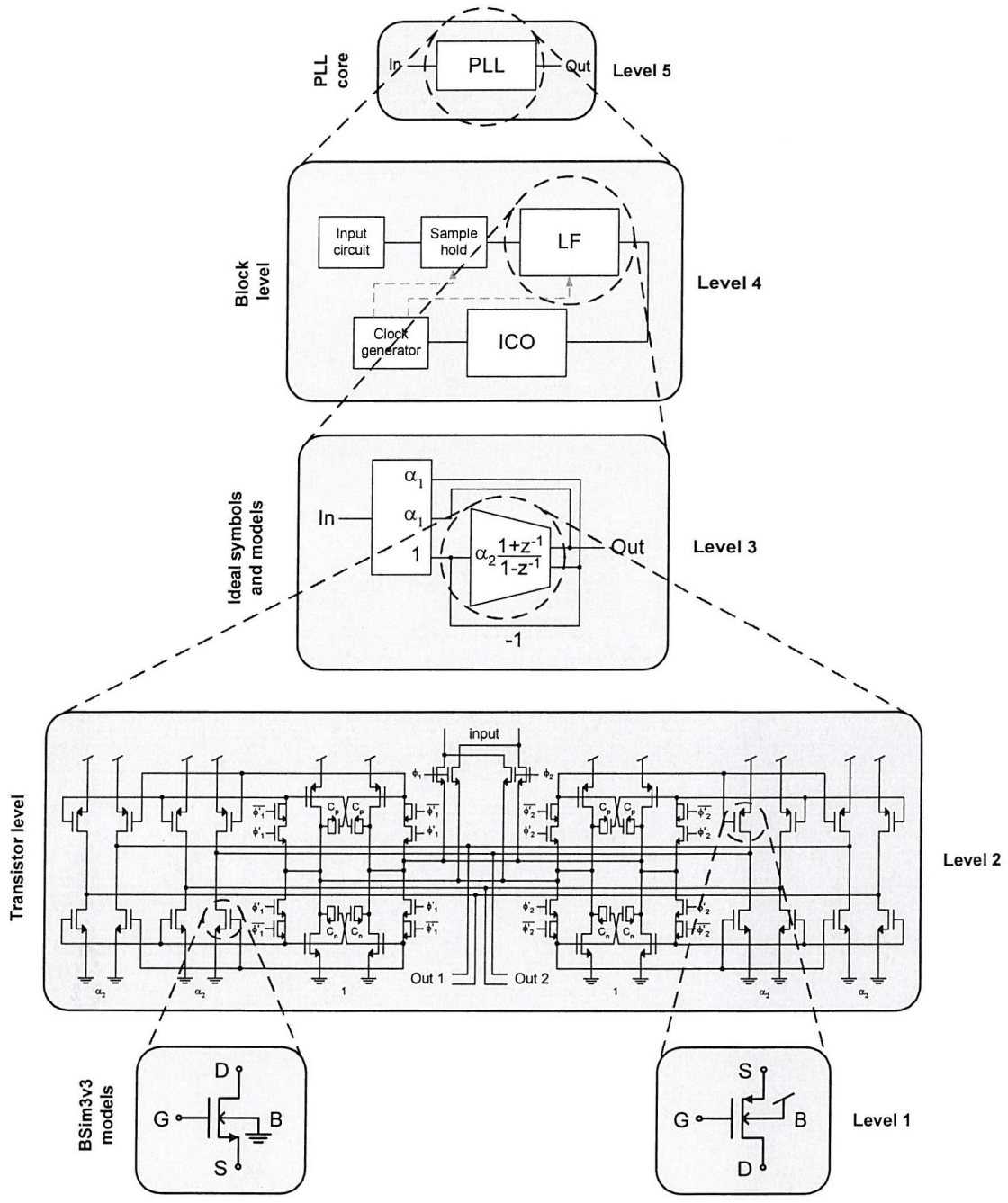


Figure 4.38 Example AutoPLL cell hierarchy

The entire AutoPLL cell library is constructed in Cadence to allow access through the AutoPLL CAD tool discussed in the next section. Bottom level transistor dimensions are passed intelligently up through the hierarchy using a combination of parameter passing statements, and global design variables.

4.4.2 AutoPLL CAD tool

The AutoPLL CAD tool follows the outline proposed by the CAD methodology presented in Chapter 3, Section 3.2.3, and the resulting three step flow is shown in Figure 4.39. Ideal loop design is considered in step 1 followed by transistor level design and simulation based optimisation in step 2 and verification of the entire transistor level PLL in step 3. The tool is written in SKILL®, the language of Cadence DFII, which leads to some unique advantages. AutoPLL can be invoked seamlessly from within a normal Cadence design flow and can access the AutoPLL cell library to assert design variables and instance parameters. Furthermore, the powerful Cadence simulation tool, Spectre, can be invoked from within AutoPLL and the results from simulations used as part of the design process. The tool allows simple technology retargeting by using a process file containing a handful of process parameters to drive the first cut design. The technology can be simply changed by creating a new process file.

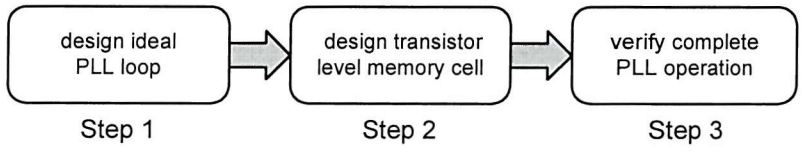


Figure 4.39 AutoPLL CAD tool design flow

Step 1 of the AutoPLL tool is shown in Figure 4.40 and involves the ideal design of the PLL loop as described in Section 4.2. The loop is defined by the PLL centre frequency, the tracking range, the loop natural frequency and damping factor. The PD gain parameter is calculated using the theory outlined in Section 4.2.1.2 and pre-designed ICO blocks available in the AutoPLL cell library allow a selection of ICO gains. The two filter coefficients, α_1 and α_2 are calculated dynamically as the slide bars for the inputs to the tool are changed, allowing real time feedback of the design feasibility. A message field is updated with a summary of the PLL design, detailing the input current required for the tracking range and the implicit PD gain. A suggested memory cell bias current is also given.

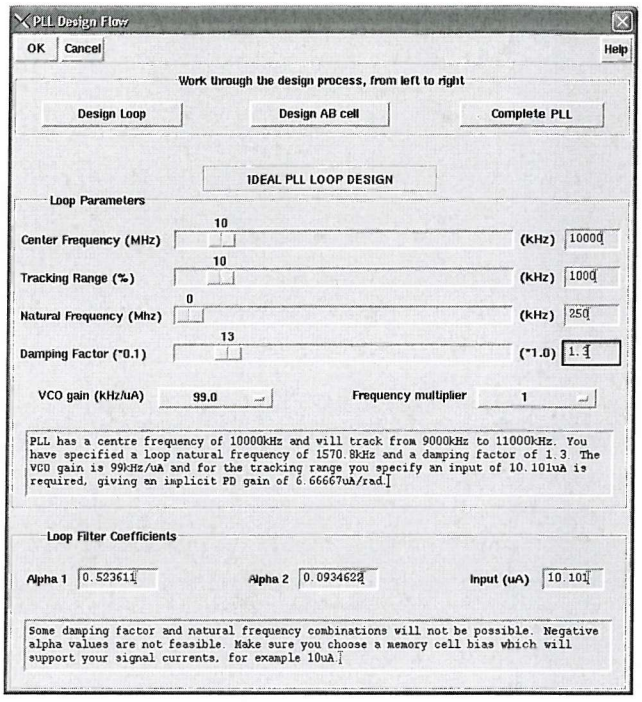


Figure 4.40 PLL loop design and filter specification (step 1)

Step 2 of the AutoPLL tool is shown in Figure 4.41 and facilitates transistor level design of the SI class AB memory cell according to the procedure in Section 4.3.1. The input specifications include sampling frequency, bias current and gate overdrive voltage, and procedures have been written to use these inputs to calculate the complete first cut memory cell design. The initial transistor dimensions are shown along with important cell parameters in the middle of the form and these are updated dynamically as the input specifications are altered. A schematic containing test circuits can be opened in order to improve the first cut design based on real simulation results. Pressing the 'Simulate first cut' button runs a DC operating point simulation for this test schematic using BSim3v3 models and automatically updates a message field in the tool window displaying key simulated quantities such as drain current and gate capacitance. The cell dimensions can be automatically optimised for a number of design parameters, including the NMOS and PMOS bias current, transconductance and gate capacitance, allowing great flexibility during the design process. A number of these parameters are inter-dependent and normally only drain current and gate capacitance would need to be optimised. Dimensions for the improved design are given in the bottom part of the form.

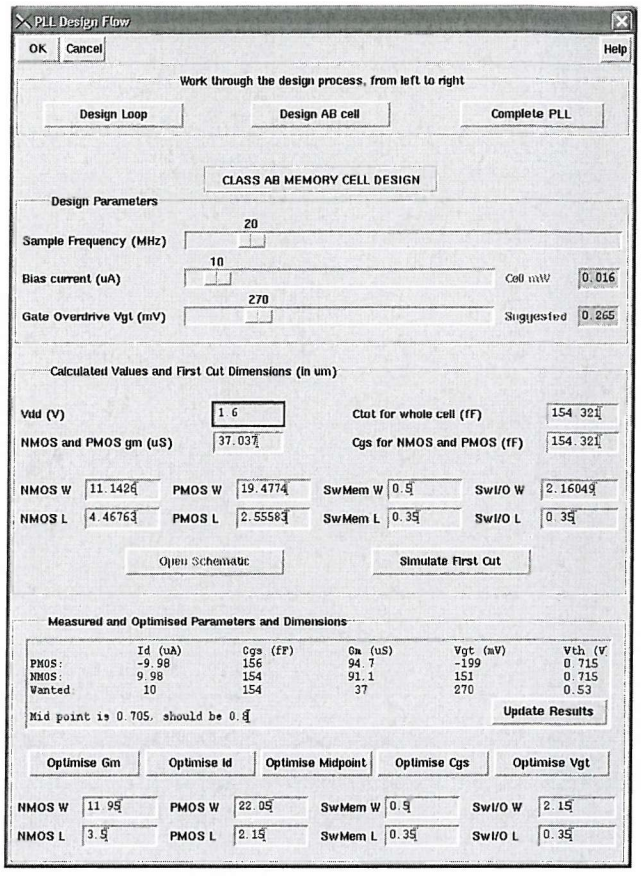


Figure 4.41 Class AB memory cell design (step 2)

Once the class AB SI memory cell has been designed, step 3 allows the entire PLL to be implemented and correct operation verified. Step 3 is shown in Figure 4.42, where it is seen that the entire state of the design process can be saved and recalled at a later stage. The schematic of the complete PLL can be opened and updated with the parameters and dimensions designed in the previous two steps. The user can then use the powerful simulation tools in Cadence to verify the correct operation of the PLL and improve the loop design if necessary.

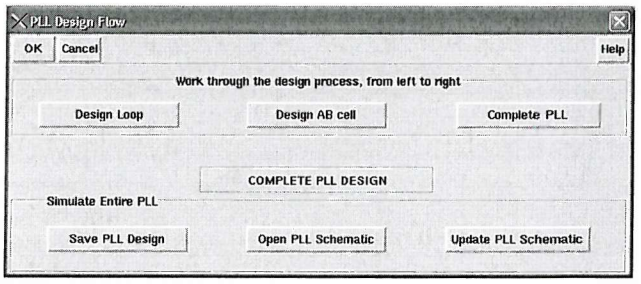


Figure 4.42 Complete SI PLL verification (step 3)

4.5 Experimental results

To validate the theoretical analysis outlined in Section 4.2 and the transistor level design of Section 4.3, experimental results from two case studies are now discussed. Both case studies demonstrate that if the correct design guidelines are followed, a separate PD is not needed for the proposed 2nd order SI PLL architecture as outlined in Section 4.2.1. The first case study is designed for a frequency shift keying (FSK) demodulation application, and the second is designed for a frequency synthesis application and is based on a state of the art PLL IP core commercially available from Barcelona Design [117].

4.5.1 Case Study 1

The first case study uses the class AB memory cell and ICO designed as examples in Sections 4.3.1 to 4.3.4 to form a SI PLL to the specifications given in Table 4.3 and is targeted towards FSK demodulation applications.

Table 4.3 Case study 1 specifications

Centre frequency:	10MHz
Input track range:	9MHz-11MHz
Output track range:	9MHz-11MHz
Loop bandwidth:	250kHz
Damping factor:	1.3
Process:	0.35μm CMOS
Output duty cycle:	50%

From Figure 4.34 the ICO gain, K_o, at zero output is read as 495kHz/μA, which, for this application would benefit from being smaller and so is reduced by a factor of 5 by using a current mirror scale stage before the oscillator in the ICO (see Section 4.3.4). This gives a K_o of 99 kHz/μA and so to achieve the specified track range the ICO input will have to extend from -10.1μA to 10.1μA. The theoretical implicit PD gain, K_d, is calculated from Eq. (4.36) as 6.66μA/rad. A desired loop bandwidth of 250kHz gives w_n as 1.571Mrad/sec and given ζ as 1.3 allows us to calculate the LF

coefficients using Eq. (4.47). Since this is not a frequency synthesis example, N is equal to 1.

$$\alpha_1 = \frac{N\omega_n}{K_o K_d}\left(2\zeta - \frac{N\omega_n}{K_o K_d}\right) = 0.518 \qquad (4.74)$$

$$\alpha_2 = \frac{N\omega_n{}^2 T}{2K_o K_d} = 0.0935 \qquad (4.75)$$

This completes the case study design, and the entire transistor level circuit for the SI PLL is shown for completeness in Figure 4.46. In order to evaluate the performance of the PLL designed in this case study, a FSK modulated signal, with a frequency shift of ±5% from 10MHz was applied to the input. The transient output of the LF (i.e. the demodulated FSK output) is shown in Figure 4.43.

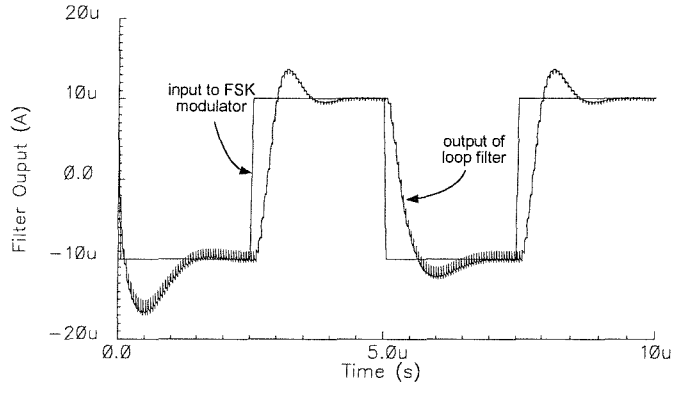


Figure 4.43 Loop filter output showing demodulation of the FSK signal

It is clear that the PLL is working as expected, and following the FSK modulations to produce a faithful output copy of the original modulating signal. It is a simple matter to adjust the damping factor, ζ, independently of the loop bandwidth, ω_n, by adjusting the SI filter coefficient α_1. This is demonstrated here by increasing α_1 to 0.9 and the resulting transient response of the demodulated FSK output is shown in Figure 4.44. The difference in overshoot is entirely expected, since the implicit PD analysis outlined in Section 4.2.1 showed that K_d was proportional to frequency. The initial fall and rise is also as expected since the signals will be out of phase to begin

with, so the PLL has to acquire lock. The double sampling nature of the system gives the benefit that the output is valid for the entire sample period and so it appears as more of a continuous time signal.

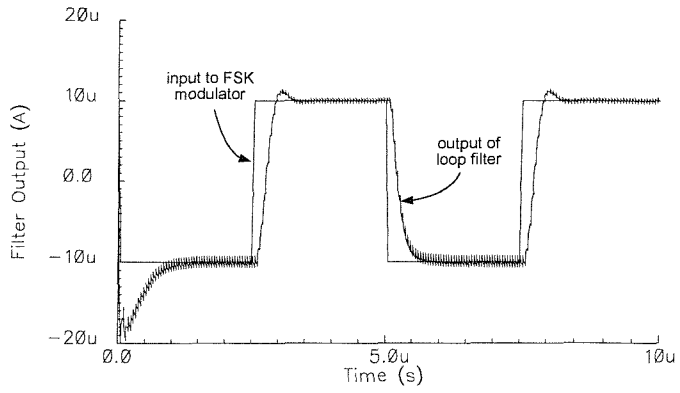


Figure 4.44 Loop filter output with increased damping factor

Figure 4.45 shows the output of the PLL ICO block as it locks to an input signal. Notice how the output locks, as expected, with a static phase error just sufficient to give an implicit PD output that maintains the ICO at the same frequency as the input. This static phase error can be seen more clearly in Figure 4.47 where two cases are shown, the first (a) is with an input frequency of -5% and (b) is with an input frequency of +5% of the centre frequency. The power consumed by this case study, including the dynamic power required by the clock generator and averaged over a complete loop acquisition period is 1.24mW. This figure is low in comparison to the 1.85mW design of [24] which operated at frequencies an order of magnitude lower than this case study.

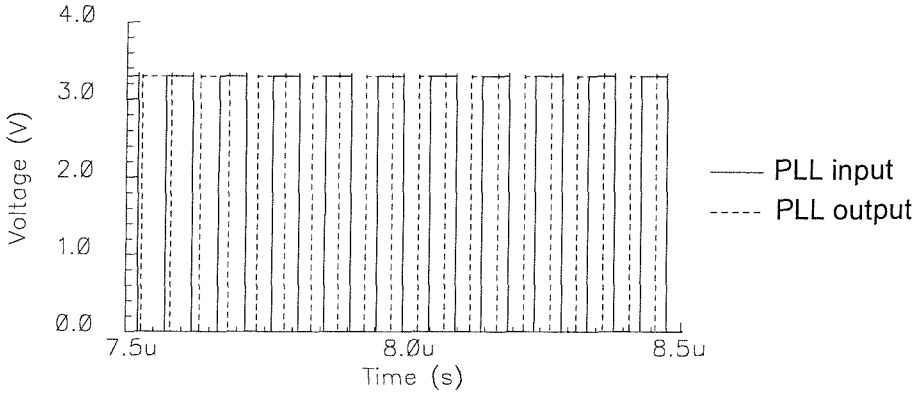


Figure 4.45 PLL output locking to an input signal

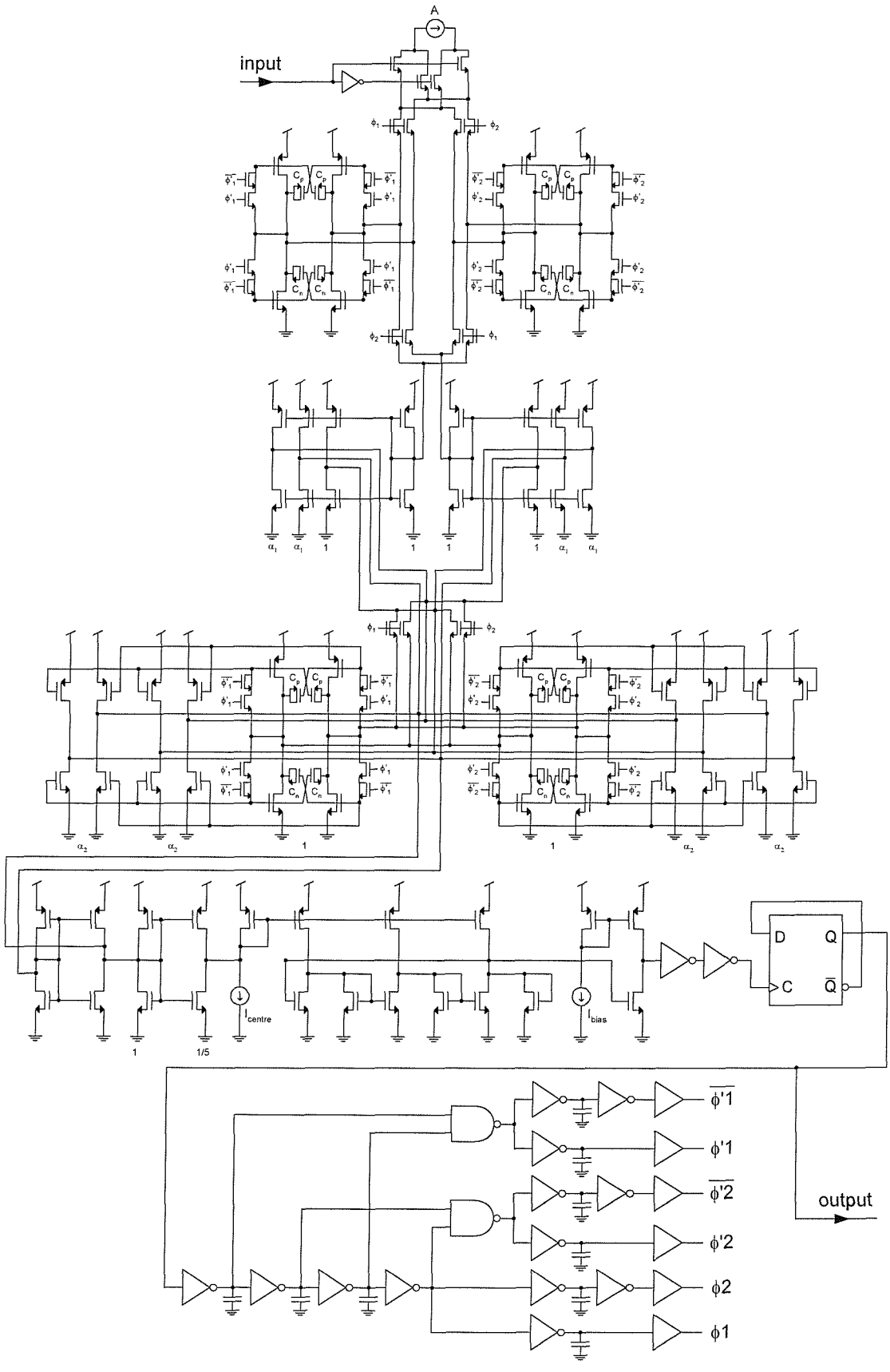


Figure 4.46 Complete SI PLL schematic

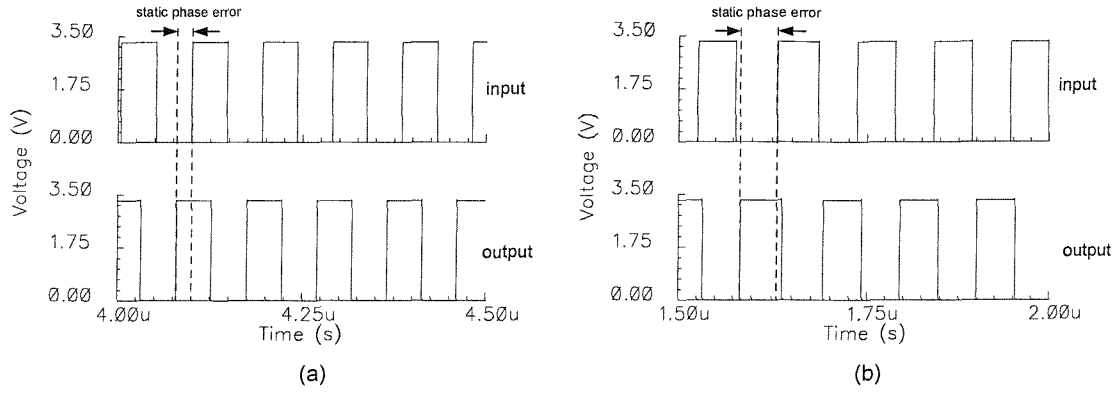


Figure 4.47 Static phase error for different input frequencies

4.5.2 Case study 2

The first case study was an example to demonstrate the design process and the performance that can be typically be expected with the proposed 2nd order SI PLL architecture. The second case study represents a more challenging example based on a state of the art PLL IP core commercially available from Barcelona Design [117]. Barcelona offer numerous PLL designs and this particular core is generated by their BCGS10-13T silicon engine which is based around the TSMC 0.13μm generic process technology. This process is far more modern than the 0.35μm process used throughout this chapter, and to achieve a similar specification using the 0.35μm process is a considerable achievement. The specifications of the Barcelona core are shown in Table 4.4 and the PLL block diagram used to achieve these specifications is shown in Figure 4.48.

Table 4.4 Barcelona design specifications

Centre frequency:	97.5MHz
Input track range:	90-105MHz
Output track range:	450MHz-525MHz
Acquisition time:	15us
Supply:	2.5V analogue, 1.2V digital
Power:	13mW
Process:	0.13μm CMOS
Output duty cycle:	50%

The Barcelona core is intended for the application of frequency synthesis, requiring an output frequency of 5 times the input frequency. As with the first case study, the ICO must be designed for twice the output frequency (10 times the input frequency) allowing a divide by 2 block to ensure a 50% duty cycle for the main output.

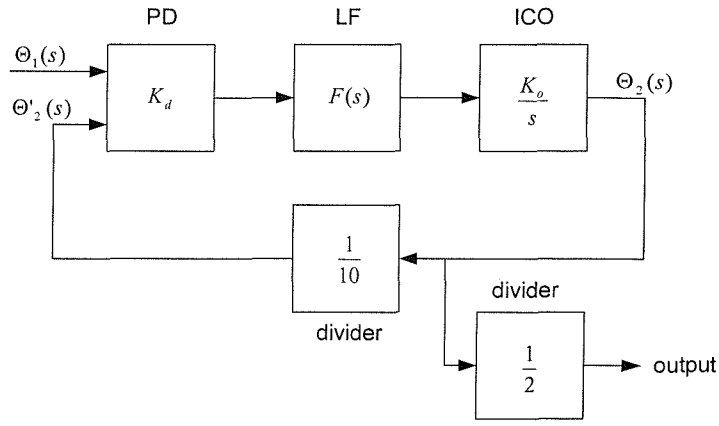


Figure 4.48 PLL block diagram for case study 2

Using the procedure outlined in Section 4.3.4 an ICO was designed of centre frequency 975MHz and output range 900-1050MHz, and the resulting transistor dimensions are shown in Table 4.5. The divide by 10 counter can be realised by implementing a state machine with four state variables and next state logic defined by the following relations:

$$Q_0^+ = \overline{Q_0} \tag{4.76}$$

$$Q_1^+ = Q_0\overline{Q_1}\,\overline{Q_3} + \overline{Q_0}Q_1 \tag{4.77}$$

$$Q_2^+ = \overline{Q_1}Q_2 + Q_0Q_1\overline{Q_2} + \overline{Q_0}Q_1Q_2 \tag{4.78}$$

$$Q_3^+ = \overline{Q_0}\,\overline{Q_1}\,\overline{Q_2}Q_3 + Q_0Q_1Q_2 \tag{4.79}$$

The output logic is given by Eq. (4.80) and an implementation of this state machine in standard NAND logic is shown in Figure 4.49.

$$Out = \overline{Q_2 Q_3} + \overline{Q_0 Q_1 Q_3} \qquad (4.80)$$

Table 4.5 ICO transistor dimensions for case study 2

Transistor	Dimensions (μm)
I_{centre}	237μA
I_{bias}	200μA
M1,M2	28.4/0.95
M3-M6	90/0/35
M9,M10	15.96/1.55
M11, M13, M15	10/0.35
M12, M14, M16	2.5/0.35
M7, M8	50/0.35
M17	18/0.35

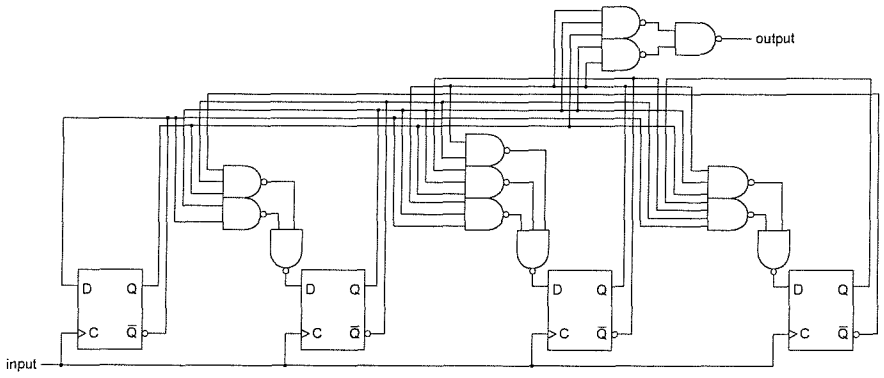


Figure 4.49 Divide by 10 counter

The transient simulation shown in Figure 4.50 confirms that the ICO, complete with divide by 10 and 2 output logic is working as expected. Furthermore the ICO performance is seen to be linear over its specified range as shown in Figure 4.51.

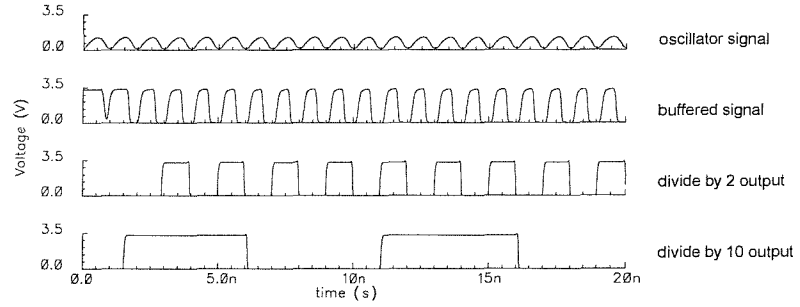


Figure 4.50 ICO and divider signals

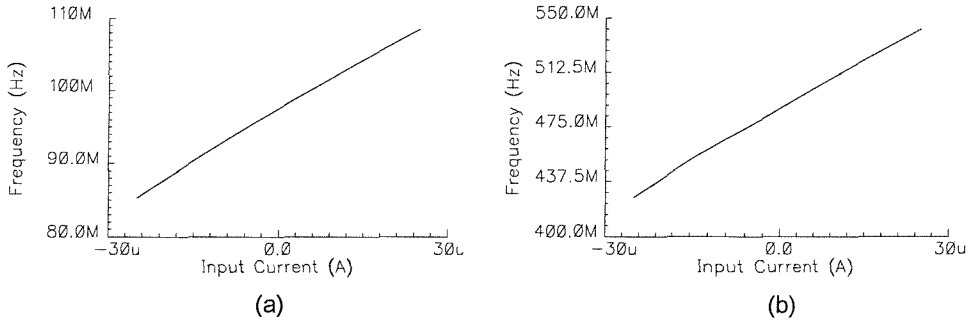


Figure 4.51 ICO performance after divide by 10 (a) and divide by 2 (b) stage

A class AB memory cell suitable for this case study was designed using the procedure in Section 4.3.1 and the resulting transistor dimensions are shown in Table 4.6. One factor which will need to be taken into account at this stage is the range of realisable filter coefficients, given that the minimum width possible for the process is 0.35μm. For the NMOS transistor, this gives a minimum scale factor of 0.0219 which is therefore the smallest value that can be assumed for either of the two filter coefficients, α_1 and α_2. This constraint will place limitations on the realisable set of loop parameters.

Table 4.6 Class AB memory cell dimensions for case study 2

Transistor	Optimised (μm)
PMOS memory w_p/l_p	28.4/0.95
NMOS memory w_n/l_n	15.95/1.55
Input/Output switch w_{si}/l_{si}	5.50/0.35
Memory switch w_{sm}/l_{sm}	0.70/0.35

The ICO gain, K_o, is found from Figure 4.51(a) to be 0.46MHz/μA and so in order to achieve the tracking range specified in Table 4.4 the input amplitude to the PD must be at least 16.3 μA. This gives a PD gain, K_d, of 10.758μA/rad (from Eq. (4.36)). A loop bandwidth of 1100kHz is chosen with a ζ of 1 and from these loop parameters the SI filter coefficients can be calculated:

$$\alpha_1 = \frac{N\omega_n}{K_o K_d}\left(2\zeta - \frac{N\omega_n}{K_o K_d}\right) = 0.843 \qquad (4.81)$$

$$\alpha_2 = \frac{N\omega_n{}^2 T}{2K_o K_d} = 0.0241 \qquad (4.82)$$

In order to evaluate whether the PLL is functioning correctly, an input with periodic frequency steps of ±5% is used and the resulting transient LF output is shown in Figure 4.52. Notice how the PLL is clearly locking to the two different frequencies and is demonstrating a less damped response for higher frequencies as predicted by the model for the implicit PD. The response is a little over damped, and this can be attributed to the greater losses in this high frequency PLL, contributing to a reduction in loop gain. Significant noise has been injected from the high frequency oscillator, which is as expected. At these frequencies, the clock edges become significant compared to the sampling period and the system begins to behave less like an ideal sampled analogue data circuit. This PLL is intended for frequency synthesis and as such would normally be locked to a fixed reference signal. Figure 4.53 shows an example of this situation, detailing the acquisition process before the loop is locked.

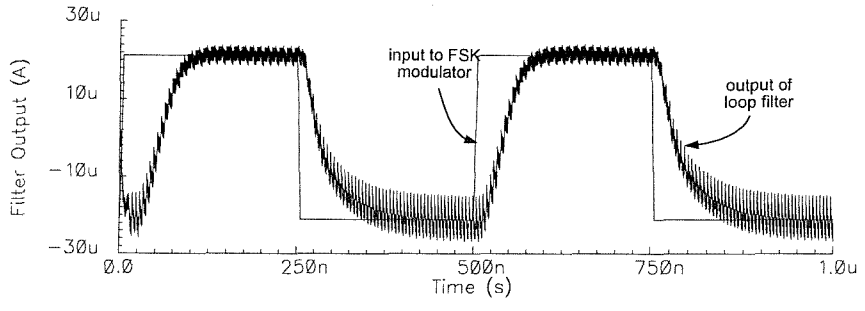


Figure 4.52 Case study 2 loop filter output with FSK input

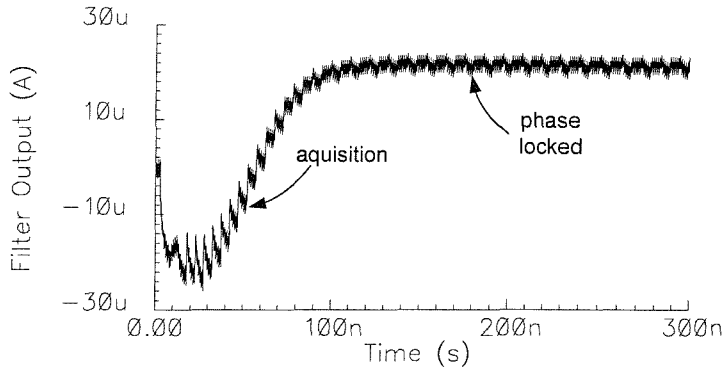


Figure 4.53 Case study 2 loop filter output, locking to a single frequency

The locking behaviour of the PLL can also be verified by the transient response shown in Figure 4.54 which shows the PLL input and the signal after the divide by 10 stage. Finally, Figure 4.55 shows the final output signal of the PLL, which is, as specified, exactly five times the frequency of the input signal. The simulated total power consumption of the PLL including the dynamic power of the logic switching elements and averaged over a typical loop acquisition cycle is 11.45mW, which is less than the Barcelona design. The low power consumption is a result of using class AB SI memory cells in the LF design. An important comparison that should be made between the SI PLL of this case study and the Barcelona design is output jitter. Unfortunately, it is very difficult to calculate jitter meaningfully using simulation and as such this would have to be left to measured results from a suitable prototype chip.

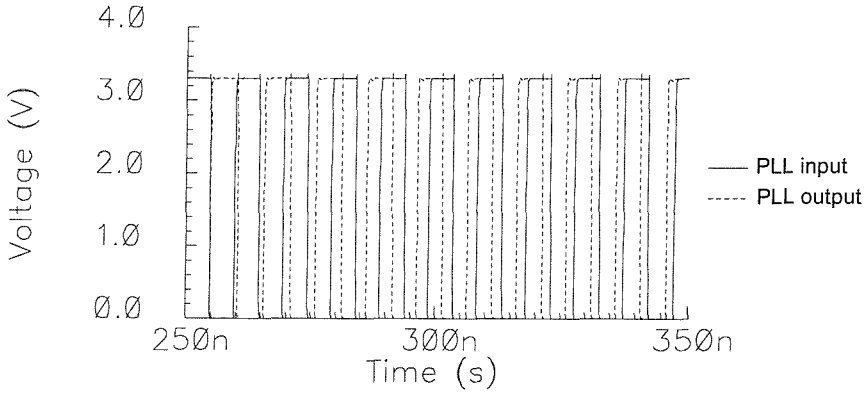


Figure 4.54 Case study 2 divide by 10 output locking to an input

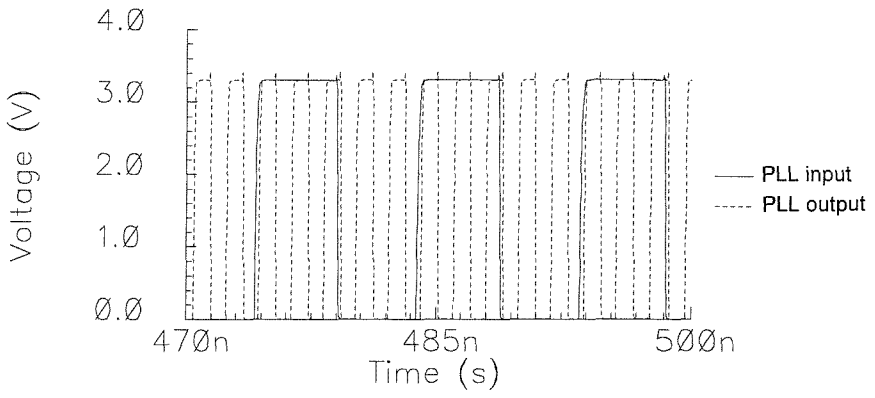


Figure 4.55 Case study 2 input and output, showing frequency synthesis

The key achievement of this case study is that a similar PLL specification previously implemented using a state of the art 0.13μm process has been realised on a 0.35μm process. Additional achievements are that the proposed SI PLL design consumes less power and requires no on-chip capacitors, which is likely to result in a very compact layout when compared to the Barcelona design [117].

4.6 Concluding remarks

This chapter has shown for the first time how the SI technique can be successfully used to implement 2nd order PLLs. It was shown in Section 4.2.1 that a separate phase detector is not necessary for the proposed SI PLL architecture. Analysis proved that instead, phase detection is implicitly achieved if the SI loop filter is clocked by the current controlled oscillator output. Furthermore, a two part model for this implicit PD has been derived and if certain guidelines are followed, its response can be approximated to a constant gain of K_d.

Careful choice and design of the SI loop filter in Section 4.2.2 led to a compact implementation and design equations have been derived relating filter coefficients to PLL loop parameters. The advantages of a SI loop filter implementation are clear: it is compatible with any standard CMOS process and requires no integrated capacitors. In Section 4.3 the transistor level design of the proposed SI PLL was outlined and detailed through a set of equations which relate high level loop parameters to low level transistor dimensions.

Two case studies have been used to demonstrate the theoretical analysis outlined in Sections 4.5.1 and 4.5.2 and include an FSK specification with 10MHz centre frequency and a more challenging specification similar to a state of the art PLL IP core commercially available. The second case study demonstrated that the proposed SI PLL architecture can support high frequency operation, even with a comparably old process technology. Furthermore, the use of the class AB SI memory cell has resulted in both case studies having very low power consumptions, which in the second case study gave a 12% power saving over the commercially available core.

Chapter 5

Conclusions and Further Research

5.1 Conclusions

As System on Chip (SoC) sets new standards in circuit integration levels, analogue designers are being faced with the challenge of implementing analogue functionality on a process optimised for digital gates. The switched-current (SI) analogue circuit technique offers a solution to this problem, since it only requires standard digital CMOS transistors and is suitable for low supply voltages. However, to unlock the full potential of this promising circuit technique, methods and tools to facilitate the design of major SI analogue building blocks are essential.

This thesis has investigated the analysis, automated design and realisation of two fundamental analogue blocks, filters and phase-locked loops (PLLs), using the SI technique. Firstly, a systematic power-aware design flow from specification to layout for SI wave filters was presented and verified through a fabricated prototype chip. Secondly a CAD methodology was developed to facilitate the rapid development of SI analogue cores and implemented for the power-aware SI wave filter design process through a tool called AutoSIF. Finally the application of the SI technique to PLL design was considered and a design flow for novel 2^{nd} order SI PLLs proposed

and automated through a further tool, called AutoPLL. A summary of the contributions of this thesis are now given.

Chapter 2 presented a complete design flow from specification to layout for SI filters using the wave synthesis technique. A key feature of the flow is a novel two stage bias and signal scaling method which not only resolves internal distortion, thus improving filter THD but also reduces power consumption as part of this process. Techniques that facilitate efficient mapping of wave adaptor coefficients to transistor level current mirrors, together with an analytical approach to the transistor level design of the main wave filter building blocks have been described. The proposed flow has been demonstrated and validated through the design, fabrication and detailed characterisation of a 3rd order lowpass elliptic filter. The filter has the highest bandwidth reported to date for fabricated SI filters using the wave synthesis technique and employs the proposed power-aware scaling approach which has resulted in a reduction in power consumption of 16.6%.

Chapter 3 proposed a generic two part CAD methodology to support the design of major SI analogue circuit functions, and an implementation of this methodology for SI wave filters in the form of AutoSIF. A carefully developed and parameterised analogue cell library and a SKILL® based CAD tool form the basis of AutoSIF, which automates the design flow and provides practical insight into SI design. Simulation based optimisation is performed at both the ideal and transistor level, leading to high performance fully defined filter schematics. AutoSIF has been used to design numerous example filters which have been validated through transistor level simulations using BSim3v3 models. AutoSIF was also used to design the fabricated 3rd order lowpass elliptic filter detailed in Chapter 2.

Chapter 4 showed the first implementation of a 2nd order PLL employing the SI technique. The proposed novel SI PLL architecture does not require a separate phase detector and instead it was shown that phase detection is implicitly achieved if the SI loop filter is sampled by the oscillator output. Guidelines have been proposed which allow the implicit phase detector relationship to be approximated to a constant gain. A compact SI loop filter has also been developed and design equations have been derived relating the PLL loop response parameters to filter coefficients. Transistor

level design of the major PLL building blocks has been achieved using low power class AB memory cells and current mirrors. The entire design flow has been automated using the generic CAD methodology proposed in Chapter 3 in the form of a cell library and tool named AutoPLL. Two case studies have been used to demonstrate the theoretical analysis and these include a 10MHz FSK demodulation example and a 500MHz frequency synthesis example similar to a state of the art PLL IP core commercially available. The operation of both case studies is verified through transistor level simulations based on BSim3v3 models. The second case study in particular demonstrated that the proposed SI PLL architecture is suitable for low power, high frequency operation.

In conclusion, this thesis has presented detailed design methods and tools for the generation of SI wave filters and PLLs. The new methods and architectures proposed are a significant contribution to the maturity of the SI technique and increase its potential for use on present and future system chips.

5.2 Further research

Based on the work in this thesis, three relevant future research directions have been identified and are briefly outlined as follows:

Class AB SI wave filters: The SI wave filter design flow in Chapter 2 and the CAD methodology in Chapter 3 have shown that it is possible to quickly and efficiently design high performance wave filters based on the S^2I memory cell, with results given for a 0.6μm process. After the work in Chapters 2 and 3 was completed, improvements were made to the newer class AB memory cell by Hughes *et al* which motivated its use in the novel PLL architecture of Chapter 4. It is possible that using class AB memory cells in wave filter designs may lead to a higher overall performance than wave filters designed using S^2I memory cells, and this possibility should be investigated. Furthermore, it is likely that the wave filter design flow proposed in Chapter 2 may well support higher frequency filters through the use of a more modern, smaller process, and this should also be the subject of further

investigation. Design of SI circuit blocks at such small feature sizes is likely to present many interesting new challenges.

Higher order SI PLL architectures: Chapter 4 demonstrated how 2nd order SI PLLs can be successfully designed using the proposed novel architecture. As with conventional 2nd order PLLs based on a lead lag loop filter, the design feasibility will depend on the specified loop natural frequency and damping factor. Some designs may indeed be entirely unfeasible or may require such a wide range of filter coefficients that the sensitivity to process variations is significantly increased. Employing higher order PLL architectures may provide a greater degree of flexibility in the design process, and although stability would have to be carefully considered, this merits further research. Furthermore, sensitivity analysis should be carried out to determine feasible PLL specification ranges for a given loop order. A number of PLL prototype chips should also be fabricated in order to directly compare their silicon area to commercially available cores, since this is likely to be a key advantage of using the SI approach.

SI PLL loop optimisation through behavioural simulation: The SI PLL architecture proposed in Chapter 4 has been validated through transistor level simulations of two case studies. Simulation of systems such as these is time intensive, with even a short transient analysis taking around an hour to complete. Also, meaningful simulated jitter and frequency domain measurements are difficult, if not impossible to obtain using standard methods. An important area of further research is therefore the development of behavioural models for SI PLL blocks, modelling not only ideal but also limited non-ideal MOS transistor behaviour. These models would allow fast optimisation of the loop response and would be especially useful when designing higher order architectures. A PLL jitter model should also be developed to facilitate accurate jitter simulations for the proposed architectures.

References

[1] D. Edenfeld, A. B. Kahng, M. Rodgers, and Y. Zorian, "2003 Technology Roadmap for Semiconductors," in *Computer*, vol. 37, pp. 47-56, Jan 2004.

[2] S. Ohr, "PANEL: Analog Intellectual Property: Now? Or Never?," in *Proc. IEEE Design Automation Conference*, pp. 181-182, June 2002.

[3] J. D. Plummer, "Silicon MOSFETs (Conventional and Non-Traditional) at the Scaling Limit," in *Proc. Device Research Conference*, pp. 3-6, June 2000.

[4] S. Matsuzaki and I. Kondo, "Information Holding Apparatus," UK:1359105, 1972.

[5] J. B. Hughes, "Analogue Techniques for Very Large Scale Integrated Circuits," Ph.D. Thesis, *Electronics and Computer Science Department, University of Southampton*, 1992.

[6] J. B. Hughes, N. C. Bird, and I. C. Macbeth, "Analogue Sampled-Data Signal Processing for VLSI Using Switched Currents," in *Proc. IEE Colloquium on Current Mode Analogue Circuits*, pp. 1-4, Feb 1989.

[7] J. B. Hughes, N. C. Bird, and I. C. Macbeth, "Switched-Currents - A New Technique for Analog Sampled-Data Signal Processing," in *Proc. IEEE International Symposium on Circuits and Systems*, pp. 1584-1587, May 1989.

[8] J. B. Hughes, I. C. Macbeth, and D. M. Pattullo, "Switched-Current Filters," *IEE Proceedings, Pt. G*, vol. 137, no. 2, pp. 156-162, April 1990.

[9] G. W. Roberts and A. S. Sedra, "Synthesizing Switched-Current Filters by Transposing the SFG of Switched-Capacitor Filter Circuits," *IEEE Transactions on Circuits and Systems*, vol. 38, no. 3, pp. 337-340, March 1991.

[10] A. Rueda, A. Yufera, and J. L. Huertas, "Wave Analogue Filters Using Switched Current Techniques," *Electronic Letters*, vol. 27, no. 16, pp. 1482-1483, August 1991.

[11] A. Yufera, A. Rueda, and J. L. Huertas, "Programmable Switched Current Wave Analog Filters," *IEEE Journal of Solid-State Circuits*, vol. 29, no. 8, pp. 927-935, August 1994.

[12] B. Jonsson and S. Eriksson, "A Low Voltage Wave SI Filter Implementation Using Improved Delay Elements," in *Proc. IEEE International Symposium on Circuits and Systems*, pp. 305-308, May 1994.

[13] J. B. Hughes, K. W. Moulding, J. Richardson, J. Bennett, W. Redman-White, M. Bracey, and R. S. Soin, "Automated Design of Switched-Current Filters," *IEEE Journal of Solid-State Circuits*, vol. 31, no. 7, pp. 898-907, July 1996.

[14] J. B. Hughes, "Top-Down Design of a Switched-Current Video Filter," *IEE Proceedings - Circuit Devices and Systems*, vol. 147, no. 1, pp. 73-81, February 2000.

[15] A. Worapishet, R. Sitdhikorn, and J. B. Hughes, "Architecture for Low-Voltage Switched-Current Complex Bandpass Filters," *Electronics Letters*, vol. 38, no. 12, pp. 535-536, June 2002.

[16] D. Macq. and P. G. A. Jespers, "A 10-Bit Pipelined Switched-Current A/D Converter," *IEEE Journal of Solid-State Circuits*, vol. 29, no. 8, pp. 967-971, August 1994.

[17] H. Matsumoto, S. Hatano, K. Tanno, Z. Tang, and O. Ishizuka, "A Switched-Current Algorithmic Analog-to-Digital Converter," in *Proc. Instrumentation and Measurement Technology Conference*, pp. 886-889, May 1994.

[18] M. Bracey, W. Redman-White, J. Richardson, and J. B. Hughes, "A Full Nyquist 15 MS/s 8-b Differential Switched-Current A/D Converter," *IEEE Journal of Solid-State Circuits*, vol. 31, no. 7, pp. 945-951, July 1996.

[19] B. E. Jonsson and H. Tenhunen, "Low-Voltage 32 MSample/s Parallel Pipelined Switched-Current ADC," *Electronic Letters*, vol. 34, no. 20, pp. 1906-1907, Oct 1998.

[20] J. Wang and C. Wey, "11-bit 4.4 mW CMOS Switched-Current D/A Converter for Low-Power/Low-Voltage Signal Processing Applications," in *Proc. Midwest Symposium on Circuits and Systems*, pp. 474-477, August 1999.

[21] J. B. Hughes, M. Mec, and W. Donaldson, "A Low Voltage 8-bit, 40 MS/s Switched-Current Pipeline Analog-to-Digital Converter," in *Proc. IEEE International Conference on Circuits and Systems*, pp. 572-575, May 2001.

[22] A. Demosthenous and J. Taylor, "A 100Mb/s, 2.8V CMOS Current-Mode Analogue Viterbi Decoder," *IEEE Journal of Solid-State Circuits*, vol. 37, no. 7, pp. 904-910, July 2002.

[23] A. Kobayashi, Y. Horio, and S. Nakamura, "State Variable Sinusoidal Switched-Current Oscillator," *Electronic Letters*, vol. 27, no. 6, pp. 489-491, March 1991.

[24] D. M. W. Leenaerts, G. G. Persoon, and B. M. Putter, "CMOS Switched Current Phase-Locked Loop," *IEE Proceedings - Circuit Devices and Systems*, vol. 144, no. 2, pp. 75-77, April 1997.

[25] A. Worspishet, R. Sitdhikorn, and J. B. Hughes, "Low-Power Complex Channel Filtering Using Cascoded Class AB Switched-Currents," in *Proc. IEEE International Symposium on Circuits and Systems*, pp. 453-456, May 2002.

[26] J. B. Hughes and K. W. Moulding, "S2I: A Switched-Current Technique for High Performance," *Electronics Letters*, vol. 29, no. 16, pp. 1400-1401, August 1993.

[27] N. C. Battersby and C. Toumazou, "Class AB Switched-Current Memory for Analogue Sampled Data Systems," *Electronics Letters*, vol. 27, no. 10, pp. 873-875, May 1991.

[28] J. B. Hughes, A. Worapishet, and C. Toumazou, "Switched-Capacitors Versus Switched-Currents: A Theoretical Comparison," in *Proc. IEEE International Symposium on Circuits and Systems*, pp. 409-412, May 2000.

[29] A. Yufera and A. Rueda, "Studying the Effects of Mismatching and Clock-Feedthrough in Switched-Current Filters Using Behavioural Simulation," *IEEE Transactions on Circuits and Systems II*, vol. 44, no. 12, pp. 1058-1067, December 1997.

[30] A. Fettweis, "Wave Digital Filters: Theory and Practice," *Proceedings of the IEEE*, vol. 74, no. 2, pp. 270-327, February 1986.

[31] A. Worapishet, P. Sirisuk, and S. Tanoi, "High-Speed Switched-Current Matched Filter for WCDMA Receivers," *Electronic Letters*, vol. 40, no. 11, pp. 639-640, May 2004.

[32] J. B. Hughes and K. W. Moulding, "An 8MHz 80Ms/s Switched-Current Filter," in *Proc. IEEE International Solid-State Circuits Conference*, pp. 60-61, Feb 1994.

[33] A. Yufera, A. Rueda, and J. L. Huertas, "Switched-Current Wave Analog Filters," in *Proc. IEEE International Symposium on Circuits and Systems*, pp. 859-862, May 1992.

[34] A. Yufera, A. Rueda, and J. L. Huertas, "A Study of the Sensitivity of Switched-Current Wave Analog Filters to Mismatching and Clock-FeedThrough Errors," in *Proc. IEEE International Symposium on Circuits and systems*, pp. 317-320, May 1994.

[35] D. Asta and D. N. Green, "Analysis of a Hybrid Analog/Switched Capacitor Phase-Locked Loop," *IEEE Transactions on Circuits and Systems*, vol. 37, no. 2, pp. 183-197, Feb 1990.

[36] R. Wilcock, P. Wilson, and B. M. Al-Hashimi, "A Novel Switched-Current Phase-Locked Loop," *Submitted to the IEEE International Symposium on Circuits and Systems*, 2005.

[37] R. Wilcock and B. M. Al-Hashimi, "Analogue Filter IP Cores for Design Reuse," in *Proc. Analog Signal Processing Conference*, pp. 2.1-2.6, November 2002.

[38] R. Wilcock and B. M. Al-Hashimi, "A CAD Methodology for Switched Current Analog IP Cores," in *Proc. IEEE International Conference on Emerging Technologies and Factory Automation Conference*, pp. 434-437, September 2003.

[39] R. Wilcock and B. M. Al-Hashimi, "Power-Conscious Design Methodology for Class-A Switched-Current Wave Filters," in *Proc. IEEE International Symposium on Circuits and Systems*, pp. 255-228, May 23-26 2004.

[40] R. Wilcock and B. M. Al-Hashimi, "Power-Aware Design Method for Class A Switched-Current Wave Filters," *IEE Proceedings - Circuit Devices and Systems*, vol. 151, no. 1, pp. 1-9, February 2004.

[41] R. Wilcock and B. M. Al-Hashimi, "Switched Current Wave Filters: from Specifications to Layout," *Submitted to the IEEE Trans. On Circuits and Systems*, 2005.

[42] N. C. Bird, "A Method of and a Circuit Arrangement for Processing Sampled Analogue Electrical Signals," UK:2213011, 1987.

[43] N. C. Bird, "Storing Sampled Analogue Electrical Currents," UK:2209895, 1987.

[44] J. B. Hughes, I. C. Macbeth, and D. M. Pattullo, "Developments in Switched-Current Filter Design," in *Proc. IEE Colloquium on Digital and Analogue Filters and Filtering Systems*, pp. 1-4, May 1990.

[45] G. Liang and D. Allstot, "FIR Filtering Using CMOS Switched-Current Technique," in *Proc. IEEE International Symposium on Circuits and Systems*, pp. 3178-3181, May 1990.

[46] T. S. Fiez, B. Lee, and D. J. Allstot, "CMOS Switched-Current Biquadratic Filters," in *Proc. IEEE International Symposium on Circuits and Systems*, pp. 2300-2303, May 1990.

[47] T. Fiez and D. Allstot, "A CMOS Switched-Current Filtering Technique," in *Proc. IEEE International Solid-State Circuits Conference*, pp. 206-207, Feb 1990.

[48] T. S. Fiez and D. J. Allstot, "CMOS Switched-Current Ladder Filters," *IEEE Journal of Solid-State Circuits*, vol. 25, no. 6, pp. 1360-1367, December 1990.

[49] A. G. Begisi, T. S. Fiez, and D. J. Allstot, "Digitally-Programmable Switched-Current Filters," in *Proc. IEEE International Symposium on Circuits and Systems*, pp. 3178-3181, May 1990.

[50] D. J. Allstot, T. S. Fiez, and G. Liang, "Design Considerations for CMOS Switched-Current Filters," in *Proc. IEEE Custom Integrated Circuits Conference*, pp. 1-4, May 1990.

[51] S.-I. Liu, C.-H. Chen, H.-W. Tsao, and J. Wu, "Realisation of IIR and FIR Filters Using Switched Current Differentiators," in *Proc. International Conference on Circuits and Systems*, pp. 688-691, June 1991.

[52] M. Song, Y. Lee, and W. Kim, "A New Clock Feedthrough Reduction Circuit in Switched-Current Filters," in *Proc. IEEE International Symposium on Circuits and Systems*, pp. 1396-1399, May 1992.

[53] B. Jonsson and S. Eriksson, "Current-Mode N-Port Adaptors for Wave SI Filters," *Electronics Letters*, vol. 29, no. 10, pp. 925-926, May 1993.

[54] B. Jonsson and S. Eriksson, "New Clock -Feedthrough Compensation Scheme for Switched-Current Circuits," *Electronics Letters*, vol. 29, no. 16, pp. 1446-1447, Aug 1993.

[55] I. Song and G. W. Roberts, "A 5th Order Bilinear Switched-Current Chebyshev Filter," in *Proc. IEEE International Symposium on Circuits and Systems*, pp. 1097-1100, May 1993.

[56] A. C. M. de Queiroz, P. R. M. Pinheiro, and L. P. Caloba, "Systematic Nodal Analysis of Switched-Current Filters," in *Proc. IEEE International Symposium on Circuits and Systems*, pp. 1801-1804, June 1991.

[57] A. C. M. de Queiroz, P. R. M. Pinheiro, and L. P. Caloba, "Nodal Analysis of Switched-Current Filters," *IEEE Transactions on Circuits and Systems II*, vol. 40, no. 1, pp. 10-18, Jan 1993.

[58] A. C. M. de Queiroz and P. R. M. Pinheiro, "Exact Design of Switched-Current Ladder Filters," in *Proc. IEEE International Symposium on Circuits and Systems*, pp. 855-858, May 1992.

[59] A. C. M. de Queiroz and P. R. M. PinHeiro, "Switching Sequence Effects in Switched-Current Filters," in *Proc. IEEE International Symposium on Circuits and Systems*, pp. 982-985, May 1993.

[60] N. C. Battersby and C. Toumazou, "Towards High Frequency Switched-Current Filters in CMOS and GaAs Technology," in *Proc. IEEE International Symposium on Circuits and Systems*, pp. 1239-1242, May 1993.

[61] N. C. Battersby and C. Toumazou, "A 5th Order Bilinear Elliptic Switched-Current Filter," in *Proc. IEEE Custom Integrated Circuits Conference*, pp. 631-634, May 1993.

[62] N. C. Battersby and C. Toumazou, "A High-Frequency Fifth Order Switched-Current Bilinear Elliptic Lowpass Filter," *IEEE Journal of Solid-State Circuits*, vol. 29, no. 6, pp. 737-740, June 1994.

[63] R. H. Zele and D. J. Allstot, "Low-Voltage Fully-Differential CMOS Switched-Current Filters," in *Proc. IEEE Custom Integrated Circuits Conference*, pp. 6.2.1-6.3.4, May 1993.

[64] R. H. Zele and D. J. Allstot, "Low-Voltage Fully Differential Switched-Current Filters," *IEEE Journal of Solid-State Circuits*, vol. 29, no. 3, pp. 203-209, March 1994.

[65] C. Toumazou, J. B. Hughes, and N. C. Battersby, *Switched-Currents: an Analogue Technique for Digital Technology*: Peter Peregrinus Ltd., 1993.

[66] A. C. M. de Queiroz and P. M. Pinheiro, "Switched-Current Ladder Band-Pass Filters," in *Proc. IEEE International Symposium on Circuits and Systems*, pp. 309-312, May 1994.

[67] J. Schechtman and A. C. M. de Queiroz, "Switched-Current Filters Using Component Simulation," in *Proc. IEEE International Symposium on Circuits and Systems*, pp. 569-572, May 1994.

[68] A. C. M. de Queiroz and P. M. Pinheiro, "Bilinear Switched-Current Ladder Filters Using Euler Integrators," *IEEE Transactions on Circuits and Systems II*, vol. 43, no. 1, pp. 66-70, January 1996.

[69] A. C. M. de Queiroz and R. R. Martins, "Experimental Bipolar Realisation of a Switched-Current Filter," in *Proc. IEEE 39th Midwest Symposium on Circuits and Systems*, pp. 18-21, August 1996.

[70] C. K. Tse and K. C. Chun, "Design of a Switched-Current Median Filter," *IEEE Transactions on Circuits and Systems II*, vol. 42, no. 5, pp. 356-359, May 1995.

[71] R. T. Goncalves, S. N. Filho, M. C. Schneider, and C. Galup-Montoro, "Programmable Switched-Current Filters Using MOSFET-Only Current Drivers," in *Proc. The 38th Midwest Symposium on Circuits and Systems*, pp. 1046-1049, August 1995.

[72] R. T. Goncalves, S. Noceti-Filho, M. C. Schneider, and C. Galup-Montoro, "Digitally Programmable Switched-Current Filters," in *Proc. IEEE International Symposium on Circuits and Systems*, pp. 258-261, May 1996.

[73] J. B. Hughes and K. W. Moulding, "A Switched-Current Double Sampling Bilinear Z-Transform Filter Technique," in *Proc. IEEE International Symposium on Circuits and Systems*, pp. 293-296, May 1994.

[74] S. Wang and M. O. Ahmad, "A New Design of Switched-Current IIR Filters," in *Proc. Canadian Conference on Electrical and Computer Engineering*, pp. 461-464, September 1995.

[75] M. O. Ahmad and S. Wang, "A Novel Fully Programmable Switched Current IIR Filter," in *Proc. IEEE Custom Integrated Circuits Conference*, pp. 267-270, May 1997.

[76] Y. L. Cheung and A. Buchwald, "A Sampled-Data Switched-Current Analog 16-tap FIR Filter With Digitally Programmable Coefficients in 0.8um CMOS," in *Proc. IEEE International Solid-State Circuits Conference*, pp. 54-55, February 1997.

[77] F. A. Farag and C. Galup-Montoro, "Digitally Programmable Switched-Current FIR Filter for Low-Voltage Applications," *IEEE Journal of Solid-State Circuits*, vol. 35, no. 4, pp. 637-641, April 2000.

[78] F. A. Farag, C. Galup-Montoro, and M. C. Schneider, "A Programmable Low Voltage Switched-Current FIR Filter," in *Proc. IEEE International Symposium on Circuits and Systems*, pp. 472-475, May 1999.

[79] A. S. de la Vega, A. C. M. de Queiroz, and P. S. R. Diniz, "Adaptive Filter Implementation Using Switched-Current Technique," in *Proc. IEEE International Symposium on Circuits and Systems*, pp. 17-20, May 2001.

[80] A. S. de la Vega, A. C. M. de Queiroz, and P. S. R. Diniz, "A Switched-Current Slice of Cells for Adaptive Filters," in *Proc. IEEE International Conference on Electronics, Circuits and Systems*, pp. 283-286, September 1998.

[81] A. C. M. de Queiroz and J. Schechtman, "Sensitivity and Error Reduction by Component Swapping in Switched-Current Filters," in *Proc. IEEE International Symposium on Circuits and Systems*, pp. 480-483, May 1999.

[82] A. C. M. de Queiroz, "Unbalanced Lattice Switched-Current Filters," in *Proc. IEEE International Symposium on Circuits and Systems*, pp. 1-4, May 2001.

[83] A. Handkiewicz, M. Kropidlowski, M. Lukowiak, and M. Bartkowiak, "Switched-Current Filter Design for Image Processing Systems," in *Proc. 13th Annual IEEE International ASIC/SOC Conference*, pp. 8-12, September 2000.

[84] A. Handkiewicz, M. Kropidlowski, and M. Lukowiak, "Computer Tools for Switched-Current Filter Design," in *Proc. The 2002 45th Midwest Symposium on Circuits and Systems*, pp. 501-504, August 2002.

[85] A. Barua and M. K. Chandrakar, "SYSCUF: Automated Synthesis of Switched Current Filter," in *Proc. IEEE International Conference on Electronics, Circuits and Systems*, pp. 999-1003, December 2001.

[86] G. Qingyun, Q. Shicai, J. Xiangluan, and S. Yonggang, "Computer Aided Design of Switched-Current Filters," in *Proc. 4th International Conference on ASIC*, pp. 94-97, October 2001.

[87] Y. Sun, *Design of High Frequency Integrated Analogue Filters*: IEE Press, 2002.

[88] R. Sitdhikorn, A. Worapishet, and J. B. Hughes, "Low-Voltage Class AB Switched-Current Technique," *Electronics Letters*, vol. 36, no. 17, pp. 1449-1450, Aug 2000.

[89] A. Worapishet, J. B. Hughes, and C. Toumazou, "Low-Voltage Class AB Two-Step Sampling Switched-Currents," in *Proc. IEEE International Symposium on Circuits and Systems*, pp. 413-416, May 2000.

[90] A. Worapishet, R. Sitdhikorn, and J. B. Hughes, "Low-Power Complex Channel Filtering Using Cascoded Class AB Switched-Currents," in *Proc. IEEE International Symposium on Circuits and Systems*, pp. 453-456, May 2002.

[91] J. D. Lancaster, B. M. Al-Hashimi, and M. Moniri, "Efficient design of switched-current lowpass elliptic wave filters using Bruton transformation," in *Proc. IEEE International Symposium on Circuits and Systems*, pp. 115 - 118, May 1998.

[92] J. D. Lancaster, B. M. Al-Hashimi, and M. Moniri, "Efficient SI Wave Elliptic Filters Based on Direct and Inverse Bruton Transformations," *IEE Proceedings of Circuits Devices and Systems*, vol. 146, no. 5, pp. 235-241, October 1999.

[93] Y. Xie, B. M. Al-Hashimi, and M. Zwolinski, "Analysis of Mirror Mismatch and Clock-Feedthrough in Bruton Transformation Switched Current Wave Filters," *IEE Proceedings - Circuit Devices and Systems*, vol. 150, no. 1, pp. 6-15, February 2003.

[94] Y. Xie and B. M. Al-Hashimi, "Analogue Adaptive Filters Using Wave Synthesis Technique," in *Proc. IEEE International Symposium on Circuits and Systems*, pp. 849-852, May 2004.

[95] C. Psychalinos and N. Kontogiannopoulos, "Switched-Current Wave Filters Based on Lossy Integrators," *Electronic Letters*, vol. 39, no. 21, pp. 1487-1488, October 2003.

[96] N. Kontogiannopoulos and C. Psychalinos, "Improved Switched Current Wave Filter Configurations Based on Microwave Prototypes," *Electronic Letters*, vol. 39, no. 11, pp. 822-823, May 2003.

[97] M. Bracey, "Current Domain Analogue-to-Digital Conversion Techniques for CMOS VLSI," Ph.D. Thesis, *Electronics and Computer Science Department, University of Southampton*, 1997.

[98] A. B. Williams, *Electronic Filter Design Handbook*: McGraw-Hill, 1981.

[99] *Affirma™ RF Simulator (SpectreRF) User Guide*: Cadence Design Systems Inc., 1999.

[100] J. B. Hughes and K. W. Moulding, "S^2I: A Two-Step Approach to Switched Currents," in *Proc. IEEE International Symposium on Circuits and Systems*, pp. 1235-1238, May 1993.

[101] P. Gray, P. Hurst, S. Lewis, and R. Meyer, *Analysis and Design of Analog Integrated Circuits*: John Wiley & Sons, 2001.

[102] P. E. Allen and D. R. Holberg, *CMOS Analog Circuit Design*: Oxford University Press, 2002.

[103] M. Ismail and T. Fiez, *Analog VLSI Signal and Information Processing*: McGraw-Hill, 1994.

[104] A. Hastings, *The Art of Analog Layout*: Prentice Hall, 2001.

[105] R. Naiknaware and T. S. Fiez, "Automated Hierarchical CMOS Analog Circuit Stack Generation with Intramodule Connectivity and Matching Considerations," *IEEE Journal of Solid-State Circuits*, vol. 34, no. 3, pp. 304-317, March 1999.

[106] I. O' Conner and A. Kaiser, "Automated Synthesis of Current-Memory Cells," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 19, no. 4, pp. 413-424, April 2000.

[107] R. A. H. Balmford, W. Redman-White, and J. B. Hughes, "Low-Distortion Interfaces and Antialias Filters for Switched-Current Applications," in *Proc. IEEE International Symposium on Circuits and Systems*, pp. 285-288, May 1994.

[108] J. B. Hughes and K. W. Moulding, "Switched-Current Signal Processing for Video Frequenceis and Beyond," *IEEE Journal of Solid-State Circuits*, vol. 28, no. 2, pp. 314-322, March 1993.

[109] J. N. Xu, J. Vital, and N. Horta, "Reusability Methodology of IC Layout - Part II," in *Proc. Design Automation and Test in Europe, Workshop*, pp. March 2001.

[110] R. Castro-Lopez, F. V. Fernandez, M. Delgado-Restituto, and A. Rodriguez-Vazquez, "Retargeting of Mixed-Signal Blocks for SoCs," in *Proc. Design Automation and Test in Europe*, pp. 772-773, March 2001.

[111] R. Castro-Lopez, F. V. Fernandez, M. Delgado-Restituto, F. Medeiro, and A. Rodriguez-Vazquez, "Generation of Technology-Portable Flexible Analog Blocks," in *Proc. IEEE International Symposium on Circuits and Systems*, pp. 61-64, May 2002.

[112] J. Wang and C. Wey, "A 12-bit 100-ns/bit 1.9-mW CMOS Switched-Current Cyclic A/D Converter," *IEEE Transactions on Circuits and Systems II*, vol. 46, no. 5, pp. 507-516, May 1999.

[113] B. E. Jonsson and H. Tenhunen, "A 3V Switched-Current Pipelined Analog-to-Digital Converter in a 5 V CMOS Process," in *Proc. International Symposium on Circuits and Systems*, pp. 351-354, May 1999.

[114] A. Tezel and T. Akin, "A Low-Power Switched-Current Algorithmic A/D Converter," in *Proc. International Symposium on Circuits and Systems*, pp. 282-285, May 1999.

[115] B. E. Jonsson and H. Tenhunen, "A Dual 3V 32-MS/s CMOS Switched-Current ADC for Telecommunication Applications," in *Proc. International Symposium on Circuits and Systems*, pp. 343-346, May 1999.

[116] D. Antonio-Torres, G. Espinosa-Flores-Verdad, A. Diaz-Sanchez, and D. Baez-Lopez, "A 1 MHz, 8-bit pipelined A/D converter using switched-current memory cells," in *Proc. IEEE Midwest Symposium on Circuits and Systems*, pp. 8-11, August 2000.

[117] Barcelona Design, http://www.barcelonadesign.com, 2004.

[118] R. E. Best, *Phase-Locked Loops Design, Simulation and Applications*, 5th ed: McGraw-Hill, 2003.

[119] P. Young, *Electronic Communications Techniques*, vol. 4th: Prentice Hall, 2002.

[120] H. C. Yang, L. K. Lee, and R. S. Co, "A Low Jitter 0.3-165 MHz CMOS PLL Frequency Synthesizer for 3 V/5 V Operation," *IEEE Journal of Solid-State Circuits*, vol. 32, no. 4, pp. 582 - 586, April 1997.

[121] A. M. Sodagar, S. M. Fakhraie, and K. C. Smith, "A Low-Voltage Current-Controlled Oscillator with Low Supply Dependency," in *Proc. Tenth International Conference on Microelectronics*, pp. 282 - 285, Dec 1998.

[122] M. A. A. El-Atta, M. A. A. El-Ela, and M. K. E. Said, "Current-Controlled Oscillator with Practical Implementation in Phase-Locked Loop," in *Proc. Nineteenth National Radio Science Conference*, pp. 494-501, March 2002.

[123] J. Dong-Youl, C. Sang-Hoon, S. Won-Chul, and C. Gyu-Hyeong, "CMOS Current-Controlled Oscillators Using Multiple-Feedback-Loop Ring Architectures," in *Proc. 44th ISSCC Solid-State Circuits Conference*, pp. 386-387, Feb 1997.

[124] H. Djahanshahi and C. A. T. Salama, "Differential CMOS Circuits for 622-MHz/933-MHz Clock and Data Recovery Applications," *IEEE Journal of Solid-State Circuits*, vol. 35, no. 6, pp. 847-855, June 2000.

[125] W. Chunyan, M. O. Ahmad, and M. N. S. Swamy, "A CMOS Current-Controlled Oscillator and its Applications," in *Proc. IEEE International Symposium on Circuits and Systems*, pp. 793-796, May 2003.

[126] D. J. Allstot, G. Liang, and H. C. Yang, "Current-Mode Logic Techniques for CMOS Mixed-Mode ASICs," in *Proc. IEEE Custom Integrated Circuits Conference*, pp. 1-4, 12-15 May 1991.

Appendix A: Prototype Chip Layout

A great deal of time and effort was invested into the design and layout of the prototype wave filter chip, described in Chapter 2, Section 2.4.5. This appendix provides further details regarding the layout of the chip, building on the stack layout strategy given in Chapter 2, Section 2.4.5. Figure A.1 shows an example of a highly matched current mirror transistor stack alongside the main layout cells from which it was built. An extensive layout cell library has been developed suitable for this purpose, allowing rapid albeit manual generation of highly matched transistor stacks.

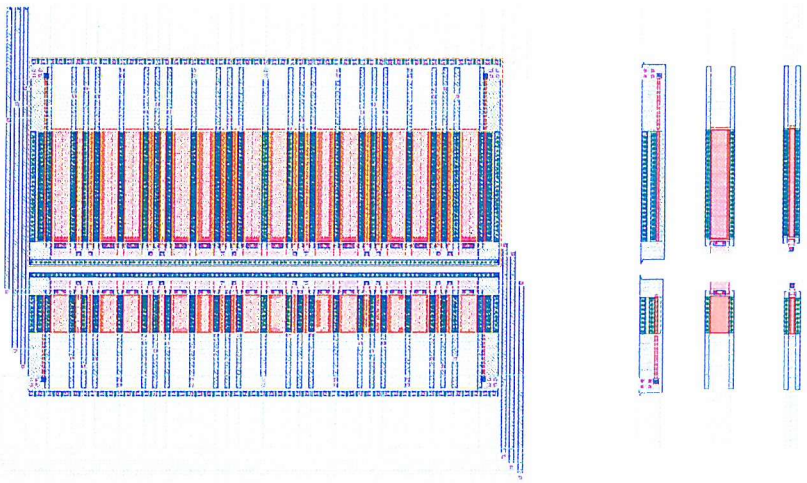


Figure A.1 Highly matched current mirror and layout cells

Figure A.2 and Figure A.3 are example layouts for the parallel and series adaptors respectively, both using current mirrors of the type in Figure A.1. Note that the power routing tracks are suitably sized not only for electromigration limitations but also for permissible voltage drops, as are the signal tracks.

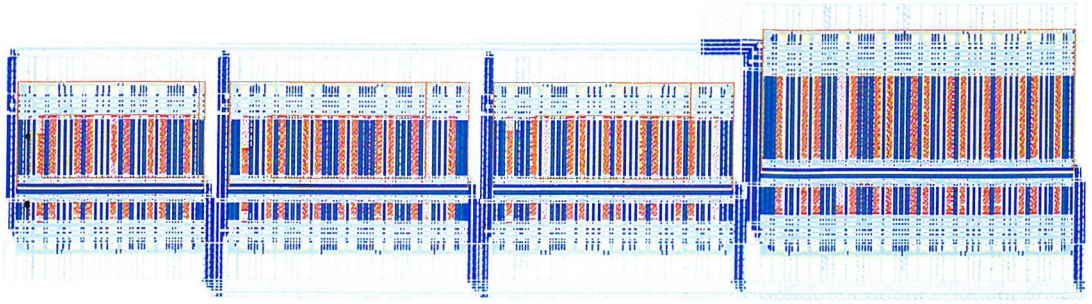


Figure A.2 Layout for parallel adaptor

Polysilicon is used exclusively for transistor gates, never for routing purposes. Apart from very few exceptions, metal1 is used for all vertical routing inside cells, with metal2 used for all horizontal routing inside and outside cells. Metal3 is used for vertical routing outside cells, since it has a much lower resistance per square than metal1 and also suffers less from cross coupling since it is physically higher up. All unusual transistor dimensions, for example those used to implement adaptor coefficients, are made from unit transistors and one non unity transistor which is always placed at the end of the stack. If a current mirror has all its transistor widths scaled, appropriately scaled layout cells are used, and hence a taller stack results.

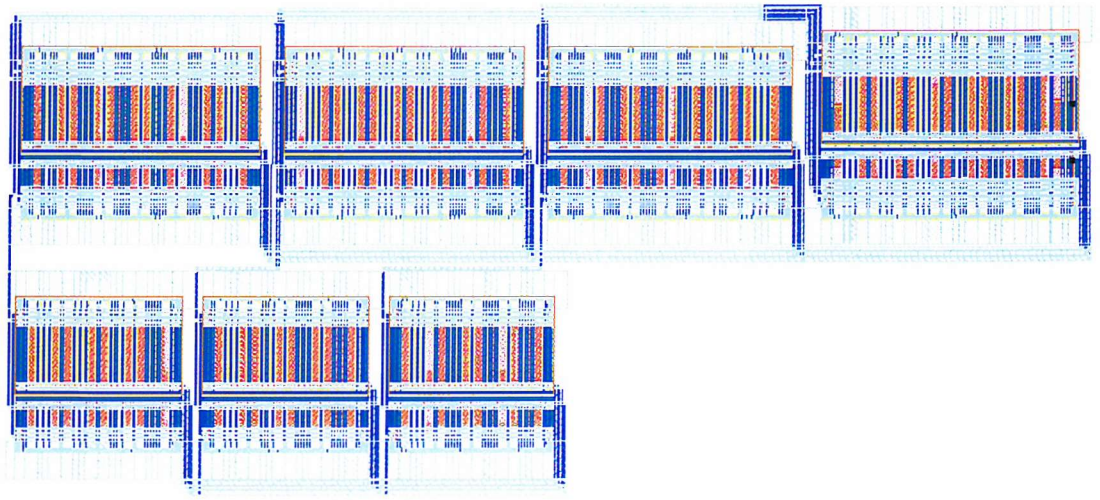


Figure A.3 Layout for series adaptor

Delay cells are laid out in two separate parts, a transistor stack, and the delay cell switches. The switches are in a physically separate area, with their own guard ring to prevent coupling of the clock signals through the substrate. This layout is shown in

Figure A.4 and the area between the group of switches and the main transistors is such that a wide substrate tap can be placed at a later stage, to further prevent digital signals coupling though the substrate into the analogue transistors.

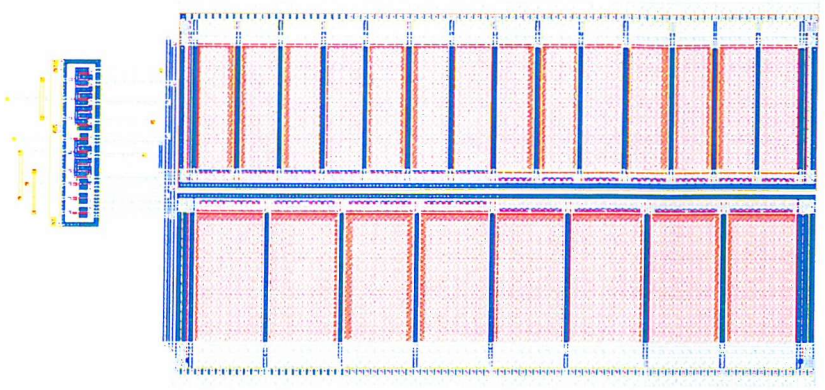


Figure A.4 Layout for delay cell with all switches in separate guard ring

The wave filter core layout consisting of the above adaptor and delay blocks is shown in Figure A.5. Delay cells are grouped on the left side such that the digital switch regions can be completely separated form the analogue transistors by a very large substrate tap tied to a quiet ground. This reduces coupling of the clock signals into the analogue sections through the substrate.

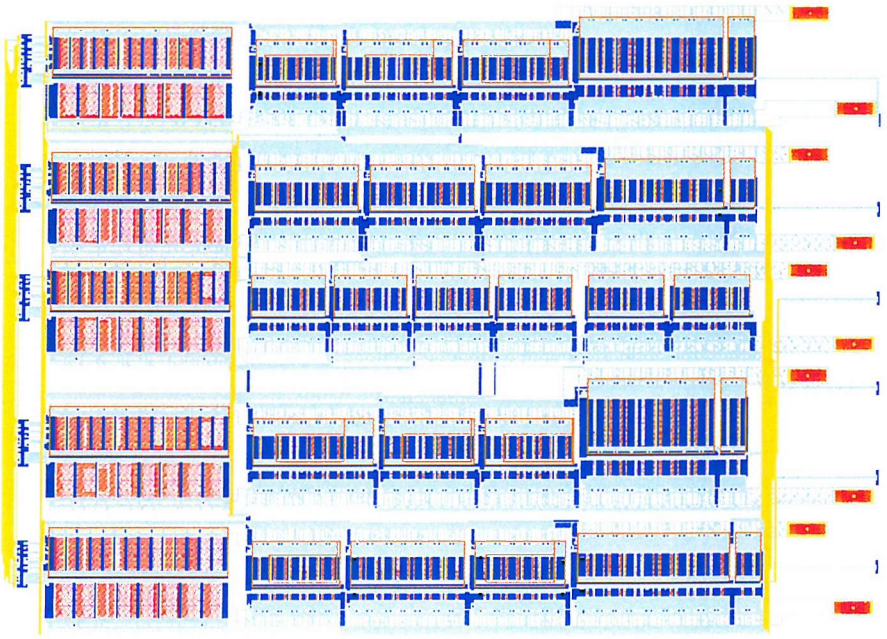


Figure A.5 Entire wave filter layout, not including bias and clock generation block

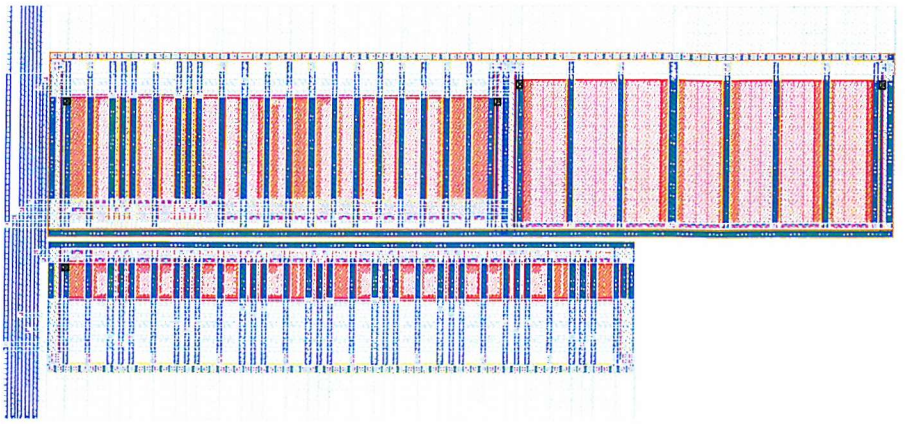


Figure A.6 Layout for bias generation block

The clock generation circuit uses standard macro cells provided by the process vendor, however, (until recently) only reduced layout cells are provided, since this protects the vendor's IP. Once the chip is taped out the fabrication house replaces instances of these reduced cells with the complete layout. The points of connection are of course given and from this the routing can be completed. The clock generation block can be seen in Figure A.7.

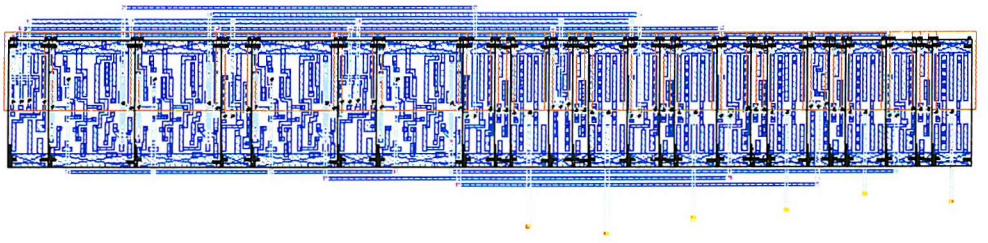


Figure A.7 Layout for clock generation block using standard cells

The entire chip layout, consisting of wave filter core, periphery and extra cells and padring is shown in Figure A.8. A number of structures were included on the chip, separate from the filter core such that individual building blocks could be characterised, should the filter fail. These extra blocks included the following:

- Clocked S^2I memory cell using the same clock phases as the wave filter
- Unclocked S^2I memory cell independently clocked and biased externally
- Two output current mirror, using same biasing as wave filter core
- Extra branches in current mirrors of adaptor blocks to indicate internal signals

The pad ring is split into an analogue pad ring and a digital pad ring, with associated separate supplies. Looking at Figure A.8 it can be seen that in fact all the purely digital signals, pads and tracks are on the very left side of the chip area, completely separate from the analogue transistors. Chip affiliation and the university logo are added as layers metal1, metal2 and metal3 for good visual contrast and to avoid large floating areas of metal these affiliations are electrically connected to the substrate. Figure A.9 shows a photo of the fabricated chip.

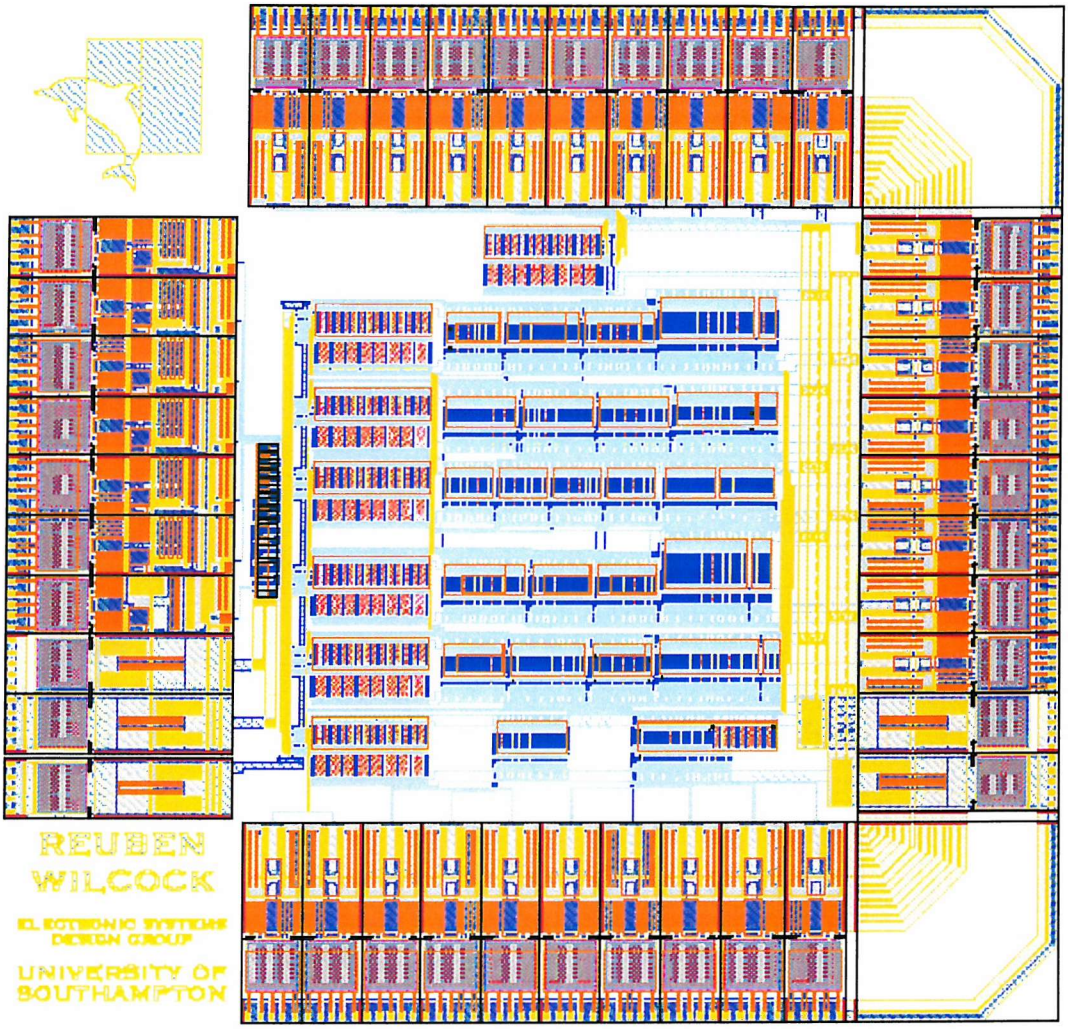


Figure A.8 Entire wave filter chip layout

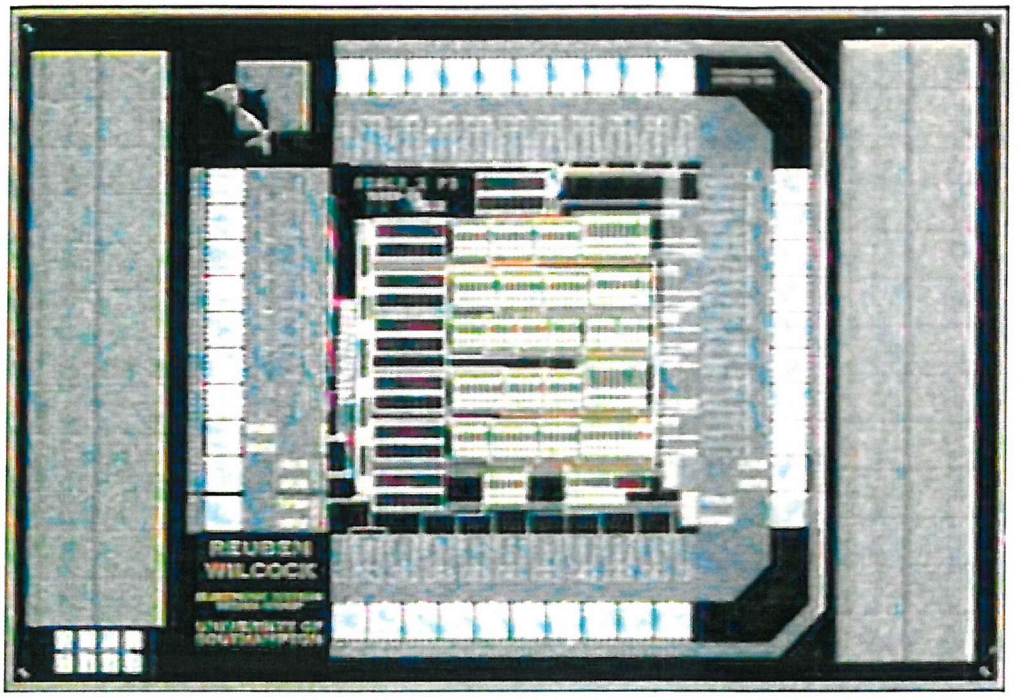


Figure A.9 Photo of fabricated die

Appendix B: Prototype Chip Results

The prototype filter chip designed and evaluated in Chapter 2 was extensively tested on return from the fabrication house. Only the most pertinent results have been shown in the main thesis, and this appendix contains a more comprehensive selection and further information detailing the test setup.

Test setup: In order to test the response of the prototype chip, an evaluation PCB was designed allowing all current outputs and inputs to be accessible as voltages (essential for use with voltage mode test equipment). The layout view of the test PCB is shown in Figure B.1. Note that all the chip pins are accessible through test points arranged and annotated around it.

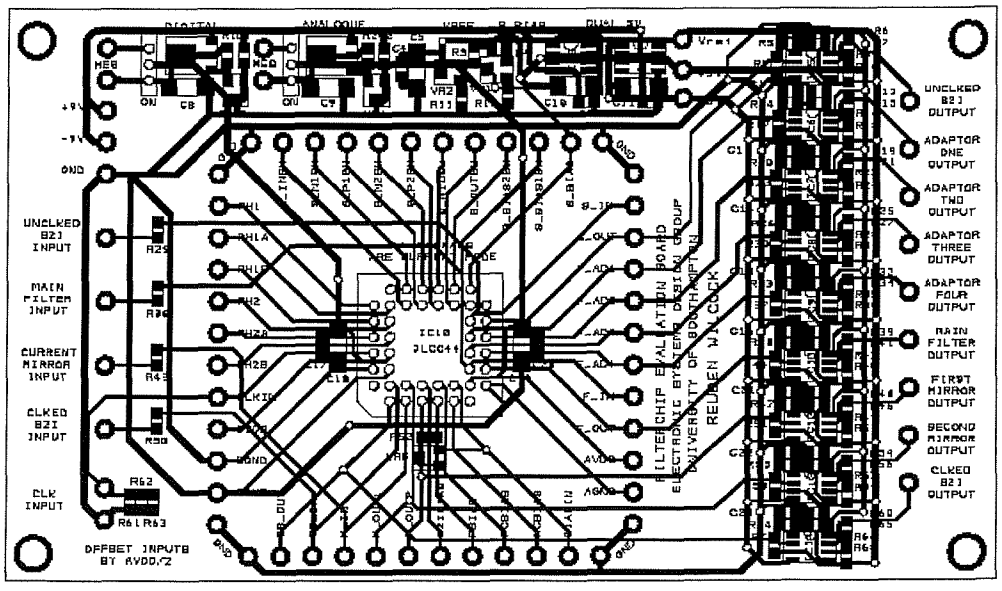


Figure B.1 Evaluation board for the prototype filter chip

The PCB is designed to fit as the top panel of a sloped plastic enclosure, which protects the bottom and provides a good base. Measurement of the analogue or digital supply voltage and current to the chip is facilitated through two power switches and additional test points. The global chip bias current is set using a multi-turn variable resistor.

The simplest and most effective means by which to convert the current signals to voltages is to use the arrangement shown in Figure B.2. A 50kΩ resistor at the input to the filter acts as a simple voltage to current converter, and is driven by a signal generator operating at an offset equivalent to the quiescent input voltage of the filter (i.e. 1.65V). A 5kΩ resistor at the output is tied to a voltage equal to the quiescent output voltage of the filter to perform current to voltage conversion. This signal is buffered (and amplified) in order to drive the capacitance of the oscilloscope and spectrum analyser cables.

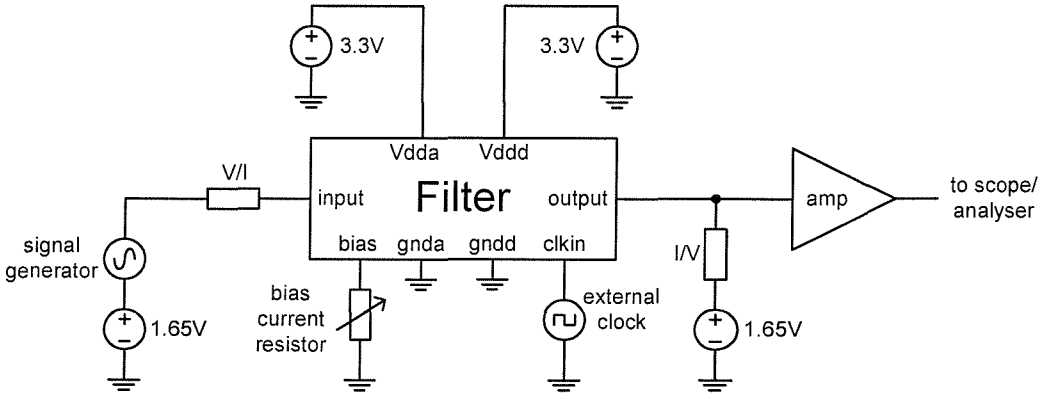


Figure B.2 Test setup for measuring filter performance

Time domain results: Figure B.3 shows the output of the filter with a sinusoidal input in the filter passband. In this case the filter sampling frequency is 10kHz. The filter output appears to be very linear, and this is reflected in later spectral measurements. Figure B.4 shows the filter output with an input signal at the filter cut-off frequency, in this case 1kHz. The sampling frequency is again 10kHz. Capturing time domain responses of the filter can be difficult at much higher frequencies since the edges of the sampled waveform contains high frequency

content. Future prototype chips will contain a sample hold stage at the filter output, to alleviate this problem.

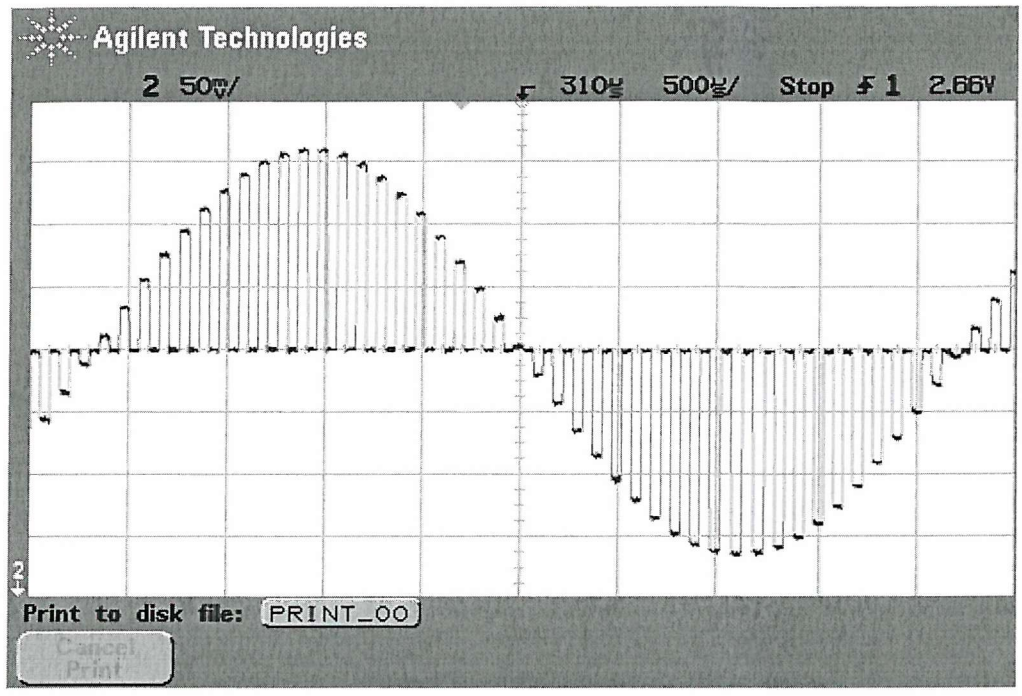


Figure B.3 Filter output with an input signal in the passband

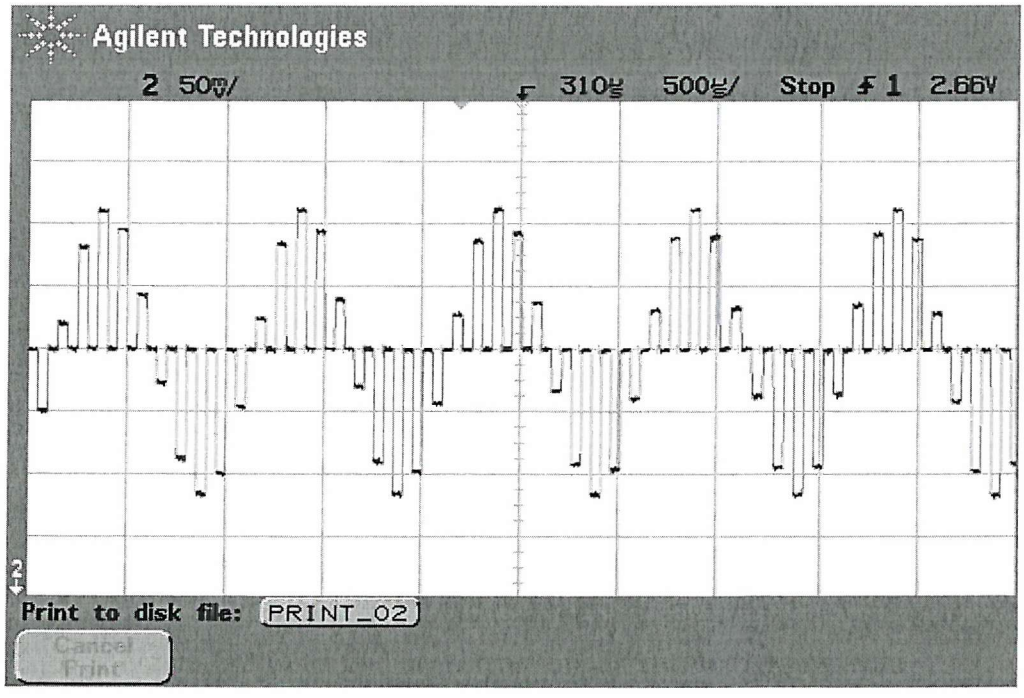


Figure B.4 Filter output with an input signal at the filter cut-off frequency

Filter frequency response: Figure B.5 shows the filter frequency response with a cut-off frequency of 100Hz (sampling frequency of 1kHz). The response demonstrates that the filter can function at much lower frequencies than the 100kHz cut-off it was designed for.

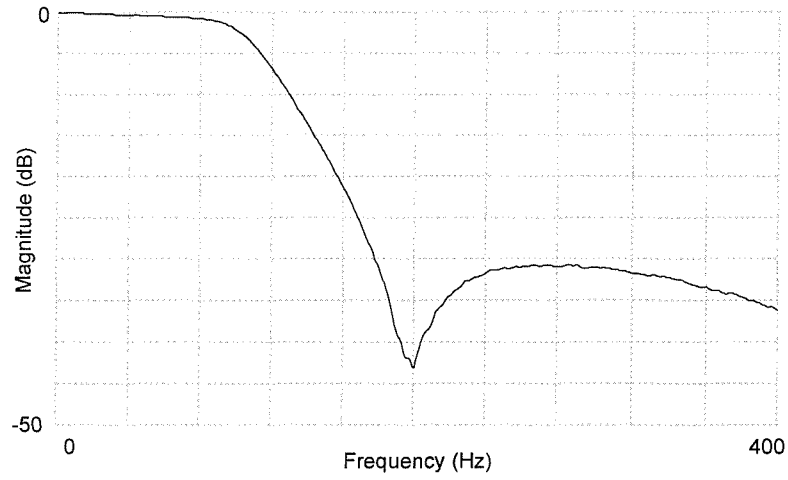


Figure B.5 Filter response with 100Hz cut-off frequency

Figures B.6 and B.7 show the filter frequency response with a cut-off frequency of 1kHz and 10kHz respectively (sampling frequency of 10kHz and 100kHz). The responses are very accurate in this frequency range and the stopband attenuation sits at -31dB. In both cases the notch depths are significant, showing accurate zero placement, which is likely to be due to the highly matched layout.

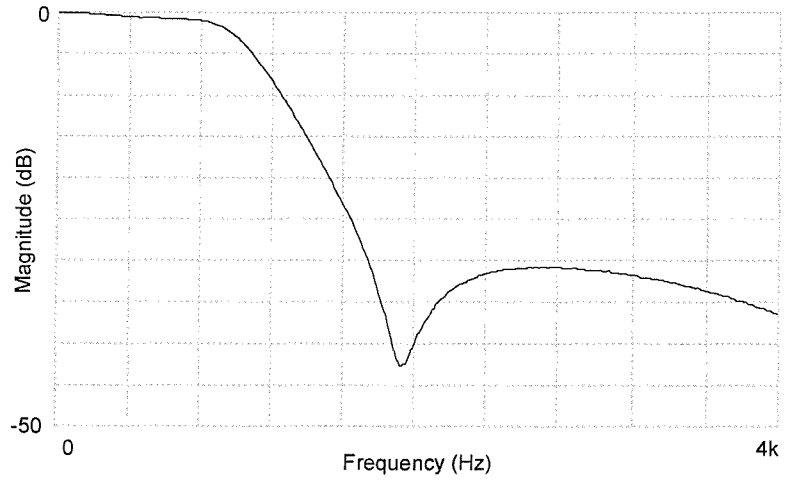


Figure B.6 Filter response with 1kHz cut-off frequency

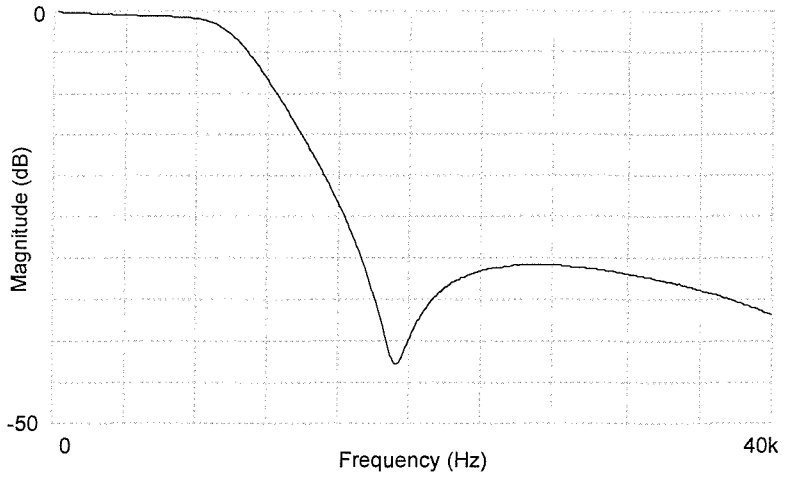


Figure B.7 Filter response with 10kHz cut-off frequency

Figure B.8 shows the filter response at the designed maximum sampling frequency of 1MHz, giving a theoretical cut-off frequency of 100kHz. The filter frequency response is close to the ideal, with a (mean) cut-off frequency of 98.2kHz and up to 0.8dB pass-band ripple compared to 0.5dB (ideal). From twenty samples, the cut-off frequency spread was found to be ±0.66% from the mean of -1.79%. It is interesting to note that the measured filter has a stopband notch at 185kHz compared to the simulated notch at 183kHz. This close agreement together with the well-defined notch of -45dB depth is attributed to the careful layout strategy.

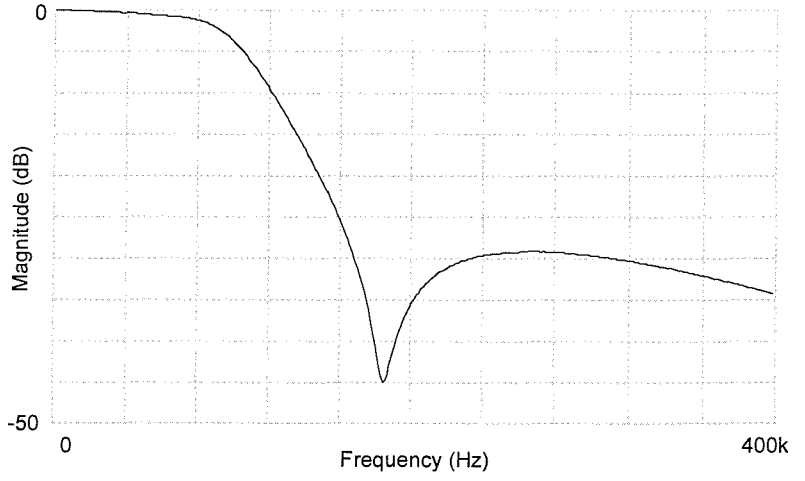


Figure B.8 Filter response with 100kHz cut-off frequency

Figure B.9 shows the effect on the filter passband of operating the filter at higher sampling frequencies, in this case 2.5MHz, giving a cut-off frequency of 250kHz. The passband ripple and stopband attenuation are worse, as expected since there will be incomplete settling in the memory cells. To demonstrate the tunability of the prototype filter, Figure B.10 shows four responses, sampled to give a cut-off frequency of 25, 50, 100 and 250kHz. The prototype chip has been successfully used to implement additional elliptic filter responses (not shown here) with cut-offs frequencies from 10Hz to 750kHz. Incomplete settling degrades the filter response at cut-off frequencies much above the designed maximum of 100kHz.

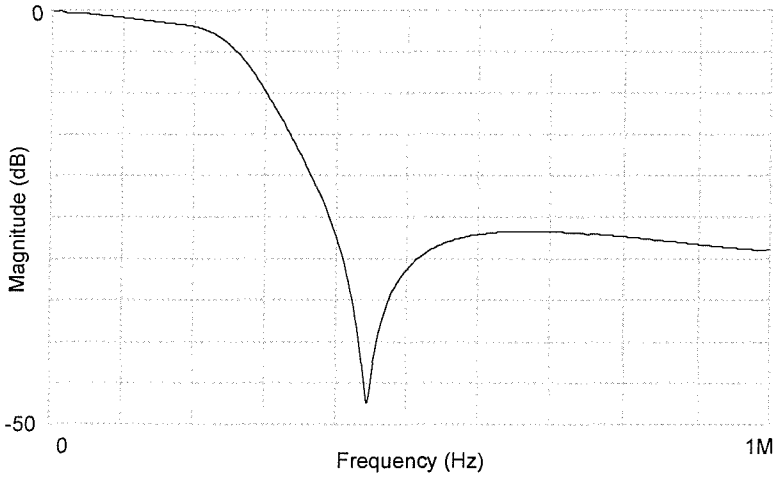


Figure B.9 Filter response with 250kHz cut-off frequency

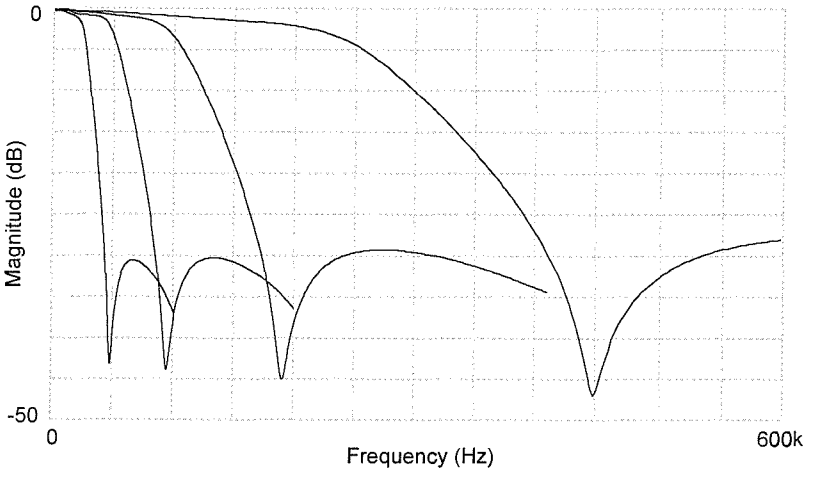


Figure B.10 Filter responses with 25, 50, 100, and 250kHz cut-off frequencies

Filter output distortion: A number of measured output frequency spectrums were plotted for various sample frequencies and input signals and these are discussed here. All spectrums were generated at an input signal amplitude of 50μA. Figure B.11 shows the spectrum resulting from an input of 1kHz to a 100kHz cut-off frequency filter. Figure B.12 shows spectrum of the output from the same filter but with an input of 10kHz.

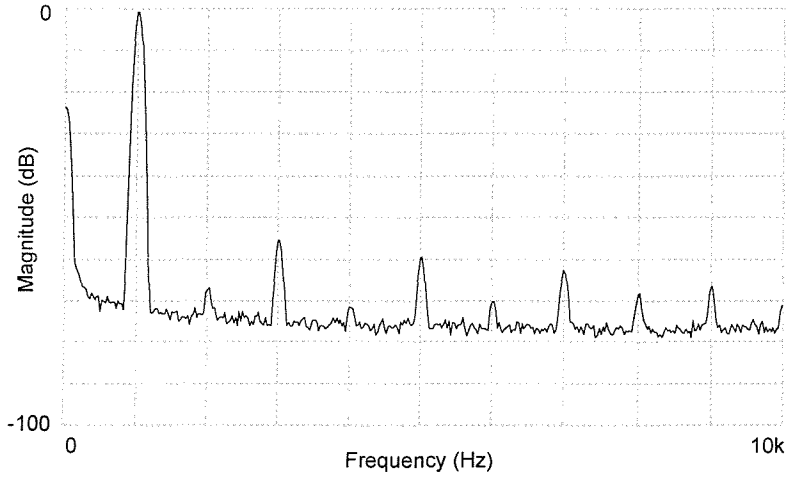


Figure B.11 Output spectrum with 1kHz input and 100kHz cut-off frequency

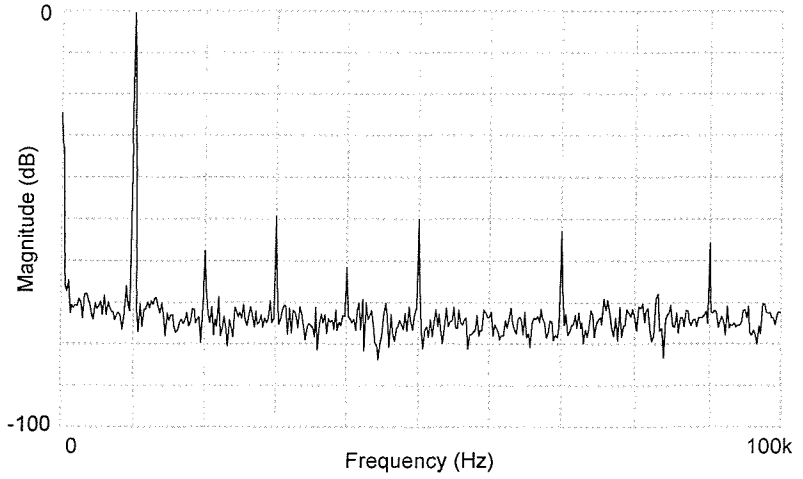


Figure B.12 Output spectrum with 10kHz input and 100kHz cut-off frequency

Figures B.13 and B.14 show the output spectrum of a 10kHz cut-off frequency filter with input signals of 100Hz and 1kHz respectively. At these lower frequencies, the harmonics are particularly small, at over 60dB less than the fundamental.

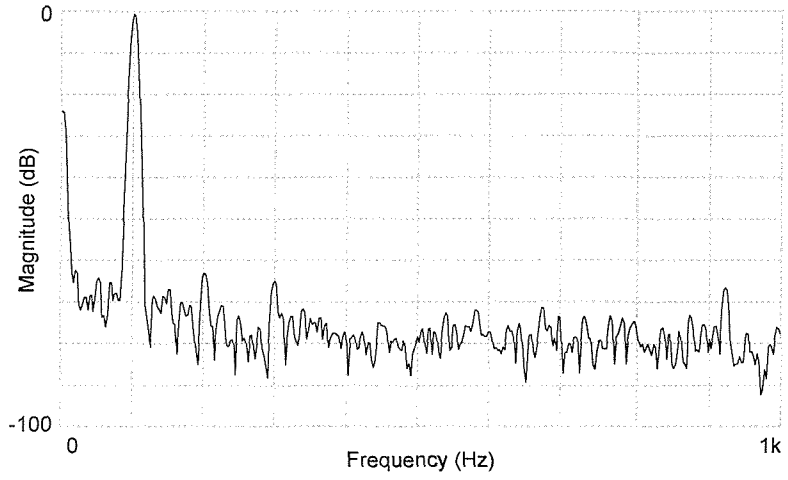


Figure B.13 Output spectrum with 100Hz input and 10kHz cut-off frequency

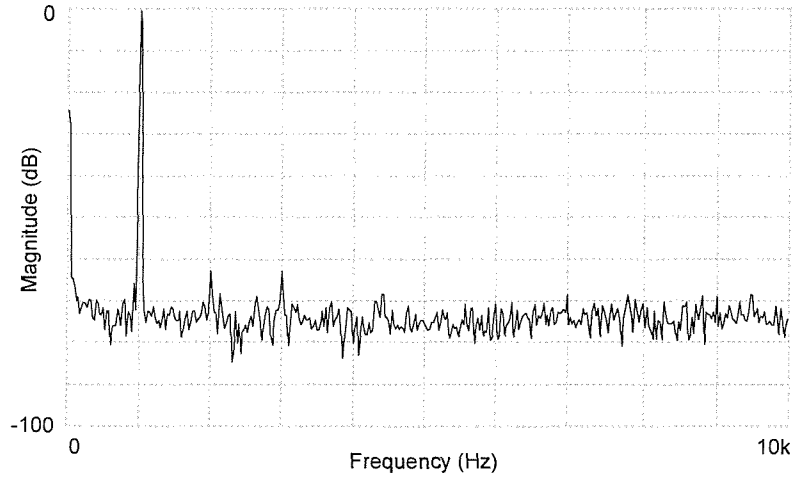


Figure B.14 Output spectrum with 1kHz input and 10kHz cut-off frequency

Filter output noise: The filter output noise was determined by observing the filter output spectrum with the chip connected and powered on, and with the input to the spectrum analyser grounded (and the chip powered off). Post-processing of the waveform determined that the integral noise levels off at about 90nA which is a little above the simulated result of 70nA, a difference which can be put down to the experimental setup. The SNR calculated from the noise power is 64dB (as compared to the simulated value of 66.62dB).

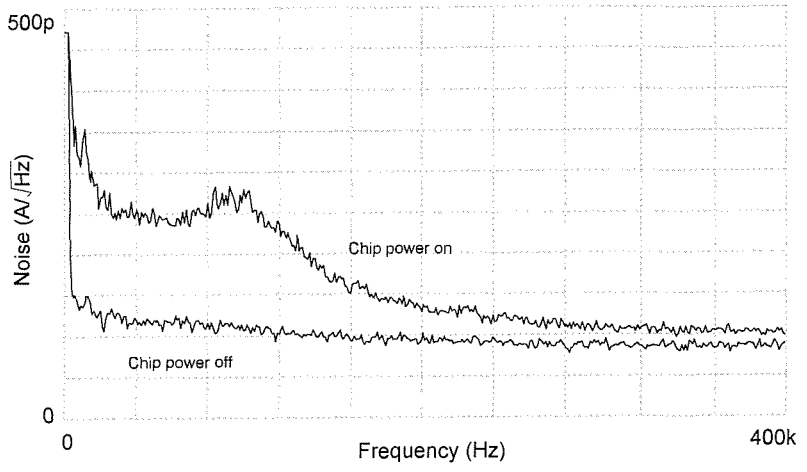
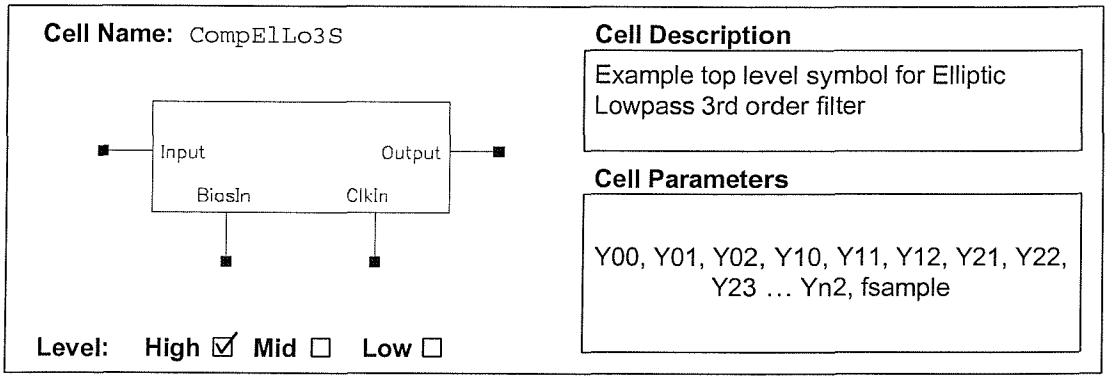


Figure B.15 Noise power spectrum with and without chip power applied

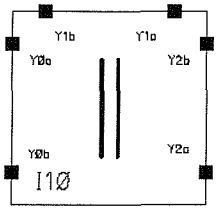
Appendix C: AutoSIF Cell Library

This appendix lists the AutoSIF cell library as discussed in Chapter 3, Section 3.3.1. A cell name ending with 'V' denotes that this is an ideal cell whereas 'R' denotes a real transistor level block. A cell name ending with an 'S' indicates a symbol which could have either ideal or real levels below it. Naming for filter cells consist of two parts, the first is either 'Pass' for passive, 'Wave' for wave ideal or 'Comp' for transistor level with the second part reflecting the filter type function and order. The AutoSIF cell library contains many different orders of Chebyshev and elliptic filters, at passive, ideal wave and transistor level. Only the 3rd order elliptic example is shown in the cell listings below.

Cell listings:



Cell Name: `ParallelS`



Level: High ☑ Mid ☐ Low ☐

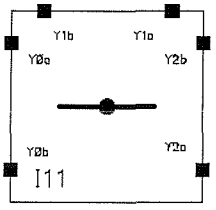
Cell Description

Parallel adaptor symbol.

Cell Parameters

Y0, Y1, Y2

Cell Name: `SeriesS`



Level: High ☑ Mid ☐ Low ☐

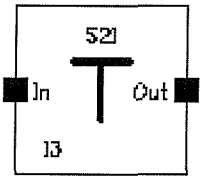
Cell Description

Series adaptor symbol.

Cell Parameters

Y0, Y1, Y2

Cell Name: `DelayS`



Level: High ☑ Mid ☐ Low ☐

Cell Description

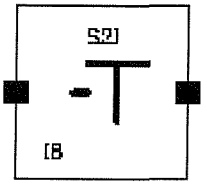
Positive delay cell symbol

Cell Parameters

None.
The delay cell design is fixed and so the transistor dimensions can be global to the entire design

---

Cell Name: `NegDelayS`



Level: High ☑ Mid ☐ Low ☐

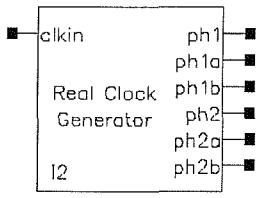
Cell Description

Negative delay cell symbol

Cell Parameters

None.
The delay cell design is fixed and so the transistor dimensions can be global to the entire design.

Cell Name: `ClockGenS`



Level: High ☑ Mid ☐ Low ☐

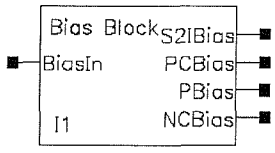
Cell Description

Symbol for clock generator block.

Cell Parameters

None.
The input sample frequency is through the terminal 'clkin'

Cell Name: `BiasBlockS`



Level: High ☑ Mid ☐ Low ☐

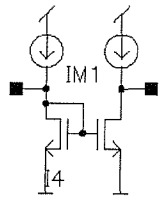
Cell Description

Symbol for bias generator block.

Cell Parameters

None.
One input bias current reference through the terminal 'BiasIn'

Cell Name: `IM1S`



Level: High ☐ Mid ☑ Low ☐

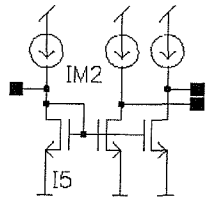
Cell Description

Single output branch current mirror symbol.

Cell Parameters

InRat, Out1Rat
Input and output scaling ratios

Cell Name: `IM2S`



Level: High ☐ Mid ☑ Low ☐

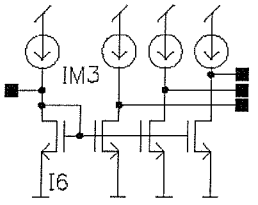
Cell Description

Double output branch current mirror smbol.

Cell Parameters

InRat, Out1Rat, Out2Rat
Input and output scaling ratios

Cell Name: IM3S



Level: High ☐ Mid ☑ Low ☐

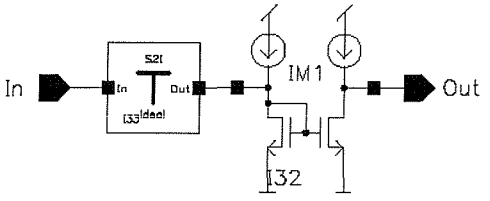
Cell Description

Triple output branch current mirror symbol.

Cell Parameters

InRat, Out1Rat, Out2Rat, Out3Rat
Input and output scaling ratios

Cell Name: NegDelayV/R



Level: High ☐ Mid ☑ Low ☐

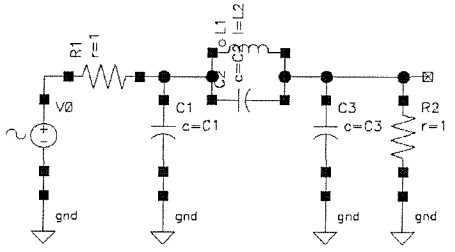
Cell Description

Negative delay cell symbolic circuit.

Cell Parameters

None.
The delay cell design is fixed and so the transistor dimensions can be global to the entire design.

Cell Name: PassElLo3



Level: High ☐ Mid ☐ Low ☑

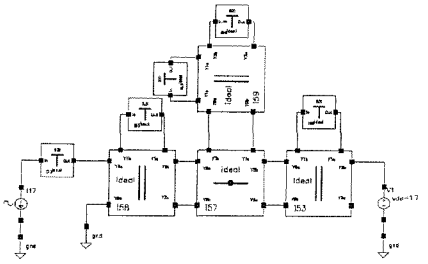
Cell Description

Example passive circuit for Elliptic Lowpass 3rd order filter.

Cell Parameters

C1, C2, C3, L2

Cell Name: WaveElLo3



Level: High ☐ Mid ☑ Low ☐

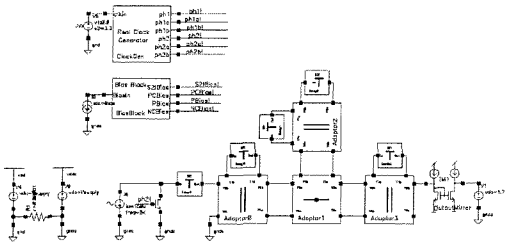
Cell Description

Example ideal wave structure for Elliptic Lowpass 3rd order filter

Cell Parameters

Y00, Y01, Y02, Y10, Y11, Y12, Y20, Y21, Y22, Y30, Y31, Y32, fsample, IBias

Cell Name: CompElLo3



Level: High ☐ Mid ☑ Low ☐

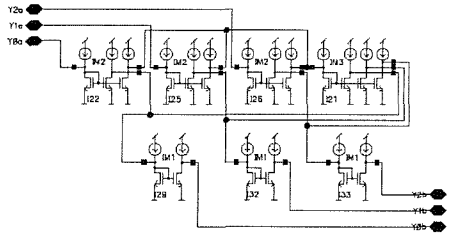
Cell Description

Example complete wave structure for Elliptic Lowpass 3rd order filter

Cell Parameters

Y00, Y01, Y02, Y10, Y11, Y12, Y20, Y21, Y22, Y30, Y31, Y32, fsample, Ibias, plus all current mirror and delay cell dimensions

Cell Name: SeriesV/R



Level: High ☐ Mid ☑ Low ☐

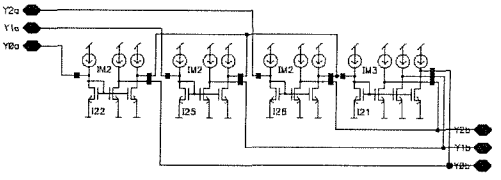
Cell Description

Symbolic implementation of series adaptor.

Cell Parameters

Y0, Y1, Y2
(adaptor coefficients)

Cell Name: ParallelV/R



Level: High ☐ Mid ☑ Low ☐

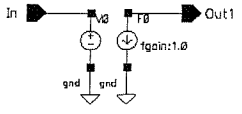
Cell Description

Symbolic implementation of parallel adaptor.

Cell Parameters

Y0, Y1, Y2
(adaptor coefficients)

Cell Name: IM1V



Level: High ☐ Mid ☐ Low ☑

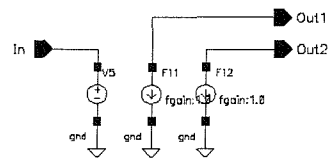
Cell Description

Ideal implementation of single output branch current mirror symbol.

Cell Parameters

Ideal current controlled current source gain factors are set by input and output branch ratios

Cell Name: IM2V



Level: High ☐ Mid ☐ Low ☑

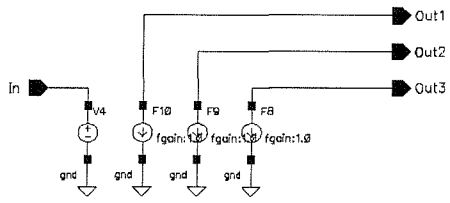
Cell Description

Ideal implementation of double output branch current mirror.

Cell Parameters

Ideal current controlled current source gain factors are set by input and output branch ratios

Cell Name: IM3V



Level: High ☐ Mid ☐ Low ☑

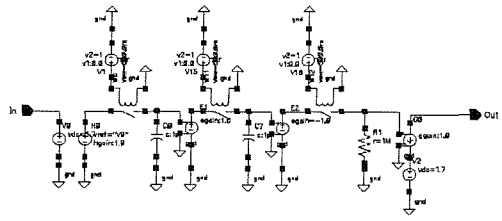
Cell Description

Ideal implementation of triple output branch current mirror.

Cell Parameters

Ideal current controlled current source gain factors are set by input and output branch ratios

Cell Name: DelayV



Level: High ☐ Mid ☐ Low ☑

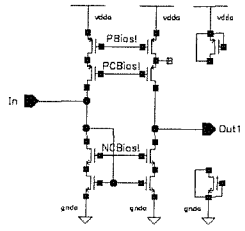
Cell Description

Ideal implementation of delay cell, using ideal switches and ideal current mirrors.

Cell Parameters

The ideal operation of the delay cell is unambiguous and so no parameters have to be changed here.

Cell Name: IM1R



Level: High ☐ Mid ☐ Low ☑

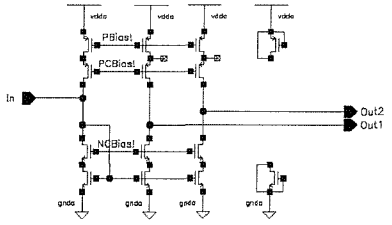
Cell Description

Transistor level implementation of single output branch current mirror symbol.

Cell Parameters

PWIn, PCWIn, NCWIn, NWIn, PWOut1, PCWOut1, NCWOut1, NWOut1, PL, PCL, NCL, NL, Dummy

Cell Name: `IM2R`



Level: High ☐ Mid ☐ Low ☑

Cell Description

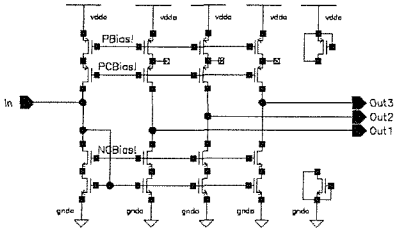
Transistor level implementation of double output branch current mirror.

Cell Parameters

PWIn, PCWIn, NCWIn, NWIn, PWOut1, PCWOut1, NCWOut1, NWOut1, PWOut2, PCWOut2, NCWOut2, NWOut2, PL, PCL, NCL, NL, Dummy

---

Cell Name: `IM3R`



Level: High ☐ Mid ☐ Low ☑

Cell Description

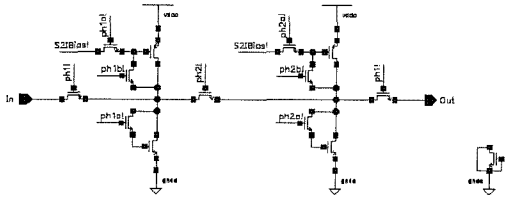
Transistor level implementation of triple output branch current mirror.

Cell Parameters

PWIn, PCWIn, NCWIn, NWIn, PWOut1, PCWOut1, NCWOut1, NWOut1, PWOut2, PCWOut2, NCWOut2, NWOut2, PWOut3, PCWOut3, NCWOut3, NWOut3, PL, PCL, NCL, NL, Dummy

---

Cell Name: `DelayR`



Level: High ☐ Mid ☐ Low ☑

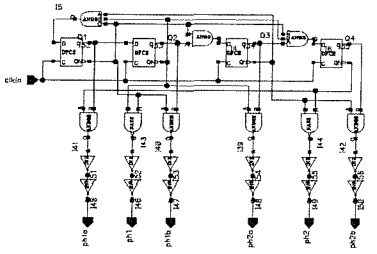
Cell Description

Transistor level implementation of delay cell, using S^2I memory cells.

Cell Parameters

S2IPW, S2IPL, S2INW, S2INL, S2IMW, S2IML, S2IIOW, S2IIOL

Cell Name: `ClockGenR`



Level: High ☐ Mid ☐ Low ☑

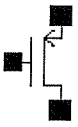
Cell Description

Transistor level implementation of clock generating block.

Cell Parameters

None.
The input sample frequency is through the terminal 'clkin. All the digital cells are from the standard libraries.

Cell Name: PMOS3



Level: **High** ☐ **Mid** ☐ **Low** ☑

Cell Description

PMOS transistor used throughout the design

Cell Parameters

W, L
Note that this simply hides the 4 terminal PMOS transistor from the standard library for easy porting between processes.

Cell Name: NMOS3

Level: **High** ☐ **Mid** ☐ **Low** ☑

Cell Description

NMOS transistor used throughout the design

Cell Parameters

W, L
Note that this simply hides the 4 terminal NMOS transistor from the standard library for easy porting between processes