Appendix A
Butterworth Filtering Transfer Function

A.1 Continuous-Time Low-Pass Butterworth Transfer Function

In order to obtain the values for the components in a filter, using the circuits transfer function, a prototype transfer function must be used. In this section it is described how to obtain the numeric values for the coefficients of the transfer function, using the Butterworth prototype transfer function. This transfer function can be obtained in several ways. It is possible to use a software tool, such as Matlab, to determine the prototype transfer function. In this case, it is necessary to indicate the order of the filter N and the desired cutoff frequency ω_p in [rad/s] (Table A.1), where n and d represent the values for the coefficients of the numerator and the denominator, respectively.

Alternatively, the filter can be designed using the mathematical equations first employed by S. Butterworth [18]. Using these equations, besides specifying the order of the filter and the cutoff frequency, it is also necessary to select the desired attenuation A_{max} at the cutoff frequency. The previous parameter is used to calculate the value of ε which is necessary to obtain the coefficients of the transfer function. The equation in Eq. A.1 shows how ε is calculated.

$$\varepsilon = \sqrt{10^{\frac{A_{max}}{10}} - 1} \tag{A.1}$$

When the order of the filter is known, one of the equations in Eq. A.2 can be used to determine the denominator of the prototype transfer function [19], where B_n is the denominator of the normalized prototype transfer function, and $\hat{S}$ is the normalized

Table A.1 Obtaining the Butterworth transfer function coefficients using Matlab for a continuous-time low-pass filter

[n,d] = butter(N,ω_p,'s')

H. A. de A. Serra, N. Paulino, *Design of Switched-Capacitor Filter Circuits using Low Gain Amplifiers,* SpringerBriefs in Electrical and Computer Engineering,
DOI 10.1007/978-3-319-11791-1

s plane.

$$\begin{cases} B_n(\hat{S}) = \prod_{k=1}^{\frac{n}{2}} \left[\hat{S}^2 - 2\hat{S}\cos\left(\pi\frac{2k+n-1}{2n}\right) + 1\right] \text{ for even n} \\ B_n(\hat{S}) = (\hat{S}+1)\prod_{k=1}^{\frac{n-1}{2}} \left[\hat{S}^2 - 2\hat{S}\cos\left(\pi\frac{2k+n-1}{2n}\right) + 1\right] \text{ for odd n} \end{cases} \tag{A.2}$$

A.1.1 First-Order Prototype Transfer Function

To implement a first-order filter, using Eq. A.2 with n = 1, the normalized prototype transfer function (Eq. A.3) is obtained.

$$T(\hat{S}) = \frac{1}{\hat{S}+1} \tag{A.3}$$

Depending on the type of filter (low-pass, band-pass, high-pass) the variable $\hat{S}$ is replaced with certain coefficients. To implement a low-pass filter, using Eq. A.3 with the low-pass coefficients, the first-order low-pass filter prototype transfer function (Eq. A.4) is obtained.

$$T\left(\hat{S} = \frac{s}{\omega_p}\varepsilon^{\frac{1}{N}}\right) = \frac{\omega_p}{\omega_p + \varepsilon^{\frac{1}{N}}s} \tag{A.4}$$

A.1.2 Second-Order Prototype Transfer Function

To implement a second-order filter, using Eq. A.2 with n = 2, the normalized prototype transfer function (Eq. A.5) is obtained. This means that the quality factor for a second-order Butterworth filter is $Q_p = 1/1.4142 = 1/\sqrt{2} = \sqrt{2}/2$.

$$T(\hat{S}) = \frac{1}{\hat{S}^2 + 1.4142\hat{S} + 1} \tag{A.5}$$

Using Eq. A.5 with the low-pass coefficients, the second-order low-pass filter prototype transfer function (Eq. A.6) is obtained.

$$T\left(\hat{S} = \frac{s}{\omega_p}\varepsilon^{\frac{1}{N}}\right) = \frac{\omega_p^2}{\omega_p^2 + \frac{\omega_p\varepsilon^{\frac{1}{N}}}{Q_p}s + \varepsilon^{\frac{2}{N}}s^2} \tag{A.6}$$

Using either software tools or the Butterworth mathematical equations, the coefficients for the prototype transfer functions can be obtained. Since usually there are more components than equations, it is necessary to choose some component values in order to extract the remaining ones.

Table A.2 Obtaining the Butterworth transfer function coefficients using Matlab for a discrete-time low-pass filter

[n,d] = butter(N,2f_p/F_s)

Table A.3 Obtaining the Butterworth transfer function poles using Matlab for a discrete-time low-pass filter

[z,p,k] = butter(N,2f_p/F_s)

A.2 Discrete-Time Low-Pass Butterworth Transfer Function

The discrete-time Butterworth prototype transfer function can be obtained in a similar way. Using Matlab, it is necessary to indicate the order of the filter N and the relation between the cutoff frequency and the clock frequency ($2f_p/F_s$) (Table A.2), where n and d represent the values for the coefficients of the numerator and the denominator, respectively.

For higher order filters that are obtained from cascading lower order filter sections the code in Table A.2 is not convenient since it returns the coefficients of the higher order filter and not the coefficients for the cascaded sections. For this case Table A.3 can be used (where z, p, and k represent the coefficients of the zeros and of the poles, and the scalar gain, respectively) to obtain the poles of each section that can then be used to obtain the coefficients of each section.

Using the Butterworth mathematical equations, in order to convert the continuous-time prototype transfer function into a discrete-time prototype transfer function, a transform (bilinear, forward Euler, backward Euler) must be used.

A.2.1 First-Order Prototype Transfer Function

Using the forward Euler transform in Eq. A.4, the prototype forward Euler transfer function (Eq. A.7) is obtained.

$$T\left(s=\frac{1-z^{-1}}{T_s z^{-1}}\right)=\frac{\omega_p T_s}{\omega_p T_s-\varepsilon^{\frac{1}{N}}+\varepsilon^{\frac{1}{N}}z} \tag{A.7}$$

When using the forward Euler transform, each continuous-time pole is converted into a discrete-time pole. Alternatively, the bilinear transform can be used, which transforms each continuous-time pole into a discrete-time pole and zero. Using this transform in Eq. A.4, the prototype bilinear transfer function (Eq. A.8) is obtained.

$$T\left(s=\frac{2}{T_s}\frac{z-1}{z+1}\right)=\frac{\omega_p T_s(1+z)}{\omega_p T_s-2\varepsilon^{\frac{1}{N}}+(\omega_p T_s+2\varepsilon^{\frac{1}{N}})z} \tag{A.8}$$

Table A.4 Obtaining the Butterworth transfer function coefficients using Matlab for a continuous band-pass filter

[n,d] = butter(N/2,[ω_L ω_H],'bandpass','s')

A.2.2 Second-Order Prototype Transfer Function

Using the forward Euler transform in Eq. A.6, the prototype forward Euler transfer function (Eq. A.9) is obtained.

$$T\left(s=\frac{1-z^{-1}}{T_s z^{-1}}\right)=\frac{T_s^2\omega_p^2}{\varepsilon^{\frac{2}{N}}+T_s^2\omega_p^2-\frac{\omega_p}{Q_p}T_s\varepsilon^{\frac{1}{N}}+\left(\frac{\omega_p}{Q_p}T_s\varepsilon^{\frac{1}{N}}-2\varepsilon^{\frac{2}{N}}\right)z+\varepsilon^{\frac{2}{N}}z^2} \tag{A.9}$$

Using the bilinear transform in Eq. A.6, the prototype bilinear transfer function (Eq. A.10) is obtained.

$$T\left((s=\frac{2}{T_s}\frac{z-1}{z+1}\right)=\frac{d(1+z)^2}{a+bz+cz^2} \tag{A.10}$$

Where,

$$\begin{cases} a=4\varepsilon^{\frac{2}{N}}+T_s^2\omega_p^2-2\frac{\omega_p}{Q_p}T_s\varepsilon^{\frac{1}{N}} \\ b=2T_s^2\omega_p^2-8\varepsilon^{\frac{2}{N}} \\ c=T_s^2\omega_p^2+2\frac{\omega_p}{Q_p}T_s\varepsilon^{\frac{1}{N}}+4\varepsilon^{\frac{2}{N}} \\ d=T_s^2\omega_p^2 \end{cases} \tag{A.11}$$

A.3 Continuous-Time Band-Pass Butterworth Transfer Function

The band-pass filter can be designed using the same methods described in Sects. A.1 and A.2. Table A.4 shows how to obtain the coefficients of the continuous time band-pass filter using Matlab, where N represents the order of the filter, ω_L and ω_H represent the lower and higher cut off frequency in [rad/s], and n and d represent the values for the coefficients of the numerator and the denominator.

Using the Butterworth mathematical equations and Eq. A.2 with n = 1, the prototype transfer function (Eq. A.12) is obtained. The prototype transfer function for a band-pass filter is designed with half the desired order since when replacing $\hat{S}$ with the band-pass coefficient, the order of the transfer function doubles.

$$T(\hat{S})=\frac{1}{\hat{S}+1} \tag{A.12}$$

Table A.5 Obtaining the Butterworth transfer function coefficients using Matlab for a discrete band-pass filter

[n,d] = butter(N/2,[$2f_L/F_s$ $2f_H/F_s$],'bandpass')

Using Eq. A.12 and replacing $\hat{S}$ with the band-pass coefficient, the second-order band-pass filter prototype transfer function (Eq. A.13) is obtained, where $Q_p = \omega_o/B$ and B is the pass band.

$$T\left(\hat{S} = \frac{Q_p\left(s^2 + \omega_o^2\right)\varepsilon^{\frac{1}{N}}}{\omega_o s}\right) = \frac{\omega_o s}{Q_p\omega_o^2\varepsilon^{\frac{1}{N}} + \omega_o s + Q_p\varepsilon^{\frac{1}{N}}s^2} \tag{A.13}$$

A.4 Discrete-Time Band-Pass Butterworth Transfer Function

Using Matlab, it is necessary to indicate the order of the filter N and the relation between the cutoff frequencies and the clock frequency ($2f_L/F_s$ and $2f_H/F_s$) (Table A.5), where n and d represent the values for the coefficients of the numerator and the denominator, respectively.

The lower and higher cutoff frequencies can be calculated using Eq. A.14.

$$\begin{cases} f_H = f_o\left(\dfrac{1}{2Q} + \sqrt{\left(\dfrac{1}{2Q}\right)^2 + 1}\right) \\ f_L = f_o\left(-\dfrac{1}{2Q} + \sqrt{\left(\dfrac{1}{2Q}\right)^2 + 1}\right) \end{cases} \tag{A.14}$$

A.4.1 Second-Order Prototype Transfer Function

Using the forward Euler transform in Eq. A.13, the prototype forward Euler transfer function (Eq. A.15) is obtained.

$$T\left(s = \frac{1 - z^{-1}}{T_s z^{-1}}\right) = \frac{\dfrac{T_s\omega_o}{Q_p\varepsilon^{\frac{1}{N}}}(z-1)}{1 + T_s^2\omega_o^2 - \dfrac{T_s\omega_o}{Q_p\varepsilon^{\frac{1}{N}}} + \left(\dfrac{T_s\omega_o}{Q_p\varepsilon^{\frac{1}{N}}} - 2\right)z + z^2} \tag{A.15}$$

Using the bilinear transform in Eq. A.13, the prototype bilinear transfer function (Eq. A.16) is obtained.

$$T\left(s = \frac{2}{T_s}\frac{z-1}{z+1}\right) = \frac{d\left(z^2 - 1\right)}{a + bz + cz^2} \tag{A.16}$$

Where,

$$\begin{cases} a = -2T_s\omega_o + 4Q_p\varepsilon^{\frac{1}{N}} + Q_pT_s^2\omega_o^2\varepsilon^{\frac{1}{N}} \\ b = -8Q_p\varepsilon^{\frac{1}{N}} + 2Q_pT_s^2\omega_o^2\varepsilon^{\frac{1}{N}} \\ c = 2T_s\omega_o + 4Q_p\varepsilon^{\frac{1}{N}} + Q_pT_s^2\omega_o^2\varepsilon^{\frac{1}{N}} \\ d = 2T_s\omega_o \end{cases} \tag{A.17}$$

Appendix B
Impulse Response Simulation and Bode Diagram Plotting

In order to simulate SC filters impulse responses, the input signal must be a rectangular pulse that allows charge into the circuit, charging capacitors. Considering that charge enters the filter during clock phase ϕ_1, the rectangular pulse must occur shortly before phase ϕ_1 is 'high' and return to 'low' shortly after ϕ_1 returns to 'low' and before ϕ_2 begins.

In order to plot the Bode diagram from the impulse response, it is necessary to obtain a sample of the impulse response in every clock period until it stops oscillating. This information is then used in Matlab's *fft* function to convert the samples into complex numbers. These complex numbers can then be used to plot the Bode diagram. Figure B.1 graphically shows the procedure to obtain the transfer function from a switched capacitor filter.

An example of the code used to plot the Bode diagram from a SC filter is shown in Table B.1.

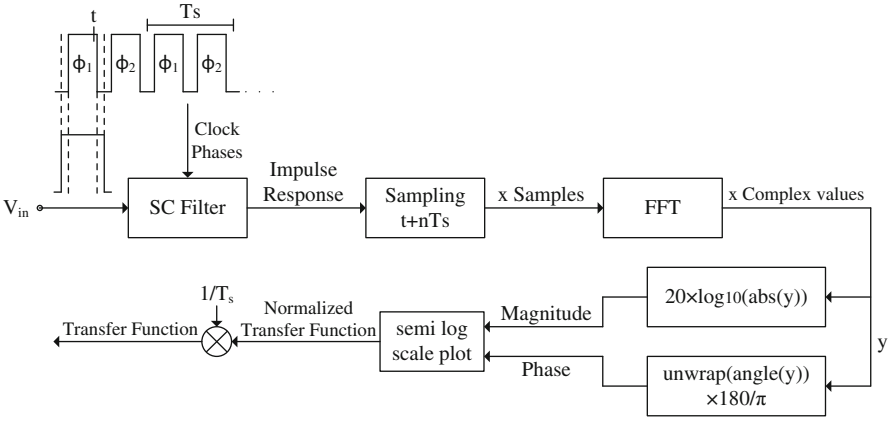

Fig. B.1 Procedure to obtain transfer function from a SC filter

H. A. de A. Serra, N. Paulino, *Design of Switched-Capacitor Filter Circuits using Low Gain Amplifiers,* SpringerBriefs in Electrical and Computer Engineering,
DOI 10.1007/978-3-319-11791-1

Table B.1 Example of how to plot the Bode Diagram

Freq = [0:n − 1]'/(n); *% n = Number of samples extracted from the impulse response*

data = **csvread**('filename.csv',1,0);

vout = data(:,2)/V; *% V = Amplitude of the input pulse*

time = data(:,1);

semilogx(Freq*Fs, 20***log10**(**abs**(**fft**(vout)))); *% Plots the magnitude*

semilogx(Freq*Fs, **unwrap**(**angle**(**fft**(vout)))*180/**pi**); *% Plots the phase*

References

1. T. Carusone, D. Johns, and K. Martin, *Analog Integrated Circuit Design*, ser. Wiley Desktop Editions. John Wiley & Sons, 2012.
2. J. T. Caves, S. D. Rosenbaum, M. A. Copeland, and C. F. Rahim, "Sampled analog filtering using switched-capacitors as resistors equivalents," in *IEEE J. Solid State Circuits*, vol. SC-12, no. 6, Dec. 1977, pp. 592–599.
3. R. Gregorian and G. Temes, *Analog MOS integrated circuitsfor signal processing*, ser. Wiley series on filters. Wiley, 1986.
4. A. P. Perez and F. Maloberti, "Performance enhanced op-amp for 65 nm CMOS technologies and below," in *Proc. IEEE Int. Symp. Circuits Systems (ISCAS'12)*, May 2012, pp. 201–204.
5. P. Allen and D. Holberg, *CMOS Analog Circuit Design*, ser. The Oxford Series in Electrical and Computer Engineering. Oxford University Press, USA, 2002.
6. B. J. Hosticka, R. W. Brodersen, and P. R. Gray, "MOS sampleddata recursive filters using switched capacitor integrators," in *IEEE J. Solid State Circuits*, vol. SC-12, no. 6, Dec. 1977, pp. 600–608.
7. K. Martin, "Improved circuits for the realization ofswitched -capacitor filters," in *IEEE Trans. Circuits and Systems*, vol. SC-27, no. 4, Apr. 1980, pp. 237–244.
8. R. P. Sallen and E. L. Key, "A practical method of designing RC active filters," in *IRE Trans. Circuit Theory*, vol. CT-2, Mar. 1955, pp. 74–85.
9. H. Serra, N. Paulino, and J. Goes, "A switched-capacitor biquadusing a simple quasi-unity gain amplifier," in *Proc. IEEE Int. Symp. Circuits Systems (ISCAS'13)*, May 2013, pp. 1841–1844.
10. M. Dessouky and A. Kaiser, "Input switch configuration suitablefor rail-to-rail operation of switched opamp circuits," in *Electronics Letters*, vol. 35, no. 1, Jan. 1999, pp. 8–10.
11. F. Michel and M. Steyaert, "A 250 mV 7.5 μW 61 dB SNDR SC $\Delta\Sigma$ modulator using near-threshold-voltage-biased inverter amplifiers in 130 nm CMOS," in *IEEE J. Solid-State Circuits*, vol. 47, no. 3, Mar. 2012, pp. 709–721.
12. B. Razavi, *Design of Analog CMOS Integrated Circuits*. New York, USA: McGraw-Hill, Inc., 2001.
13. A. D. Grasso, G. Palumbo, and S. Pennisi, "Three-stage CMOS OTA for large capacitive loads with efficient frequency compensation scheme," in *IEEE Trans. Circuits and Systems II: Express Briefs*, vol. 53, no. 10, Oct. 2006, pp. 1044–1048.
14. L. Acosta, R. G. Carvajal, M. Jimenez, J. Ramirez-Angulo, and A. Loper-Martin, "A CMOS transconductor with 90 dB SFDR and low sensitivity to mismatch," in *IEEE Int. Symp. on Circuits and Systems (ISCAS 2006)*, May. 2006, pp. 69–72.
15. F. Krummenacher and N. Joehl, "A 4-MHz CMOS continuous-timefilter with on-chip automatic tuning," in *IEEE J. Solid-State Circuits*, vol. 23, no. 3, Jun. 1988, pp. 750–758.
16. K.-C. Kuo and A. Leuciuc, ''A linear MOS transconductor usingsource degeneration and adaptive biasing," in *IEEE Trans. Circuits and Systems II: Analog and Digital Signal Processing*, vol. 48, no. 10, Oct. 2001, pp. 937–943.

H. A. de A. Serra, N. Paulino, *Design of Switched-Capacitor Filter Circuits using Low Gain Amplifiers,* SpringerBriefs in Electrical and Computer Engineering,
DOI 10.1007/978-3-319-11791-1

17. H. Serra, N. Paulino, and J. Goes, “A switched-capacitorband -pass biquad filter using a simple quasi-unity gain amplifier,” in *Technological Innovation for the Internet of Things (DoCEIS’13)*, 2013, vol. 394, pp. 582–589.
18. A. S. Sedra and K. C. Smith, *Microelectronic Circuits*, ser. The Oxford Series in Electrical and Computer Engineering. Oxford University Press, USA, 2004.
19. R. M. Golden and J. F. Kaiser, “Root and delay parameters fornormalized bessel and butterworth low-pass transfer functions,” in *IEEE Trans. Audio and Electroacoustics*, vol. AU-19, no. 1, Mar. 1971, pp. 64–71.