

# Appendix

## Time-Domain Modelling of an Electrothermal DAFL

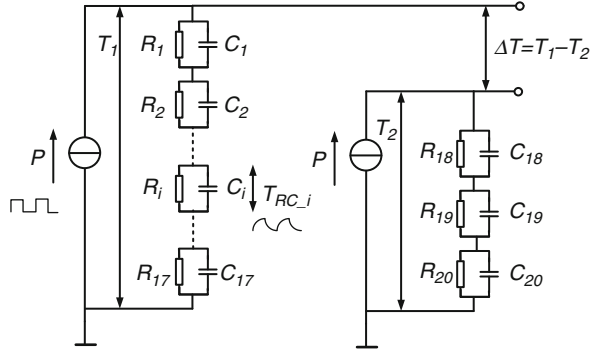
This appendix provides a time-domain modeling method for the digitally-assisted electrothermal frequency-locked loop (DAFL). A set of basic Matlab codes have been developed, which can be used in further simulations of various time-domain effects in the behavior of a DAFL. The electrothermal filter (ETF) of the loop realizes part of its signal chain in the thermal-domain. It will be shown how the analogy between the thermal and electrical domains can be used to develop a network of resistors and capacitors that reproduce the ETF characteristic. This network can be embedded in the time-domain model of the loop, which is in principle an analog mixed-mode system, with continuous-time and discrete-time functions. These can be modeled with the behavioral simulation techniques used for data converters.

### A.1 Time-Domain Simulation of an ETF

Chapter 3 provided an analytical modeling of the ETF based on its thermal impedance (see Sects. 3.3 and 3.4). This model mainly describes the frequency-domain characteristics of an ETF by means of numerical calculations, which are rather difficult to use in time-domain simulations. It should be noted that the thermal-impedance model has been extensively used in determining the various sensitivity functions that relate the DAFL performance to various error sources.

A simpler method for the time-domain modeling of an ETF is to match the frequency-domain response of a low-pass filter, e.g. a network of  $R$  and  $C$  elements (electrical-domain resistors and capacitors), to that of the ETF. This method was previously used in the development of a *wind sensor* that incorporated electrothermal filters. The step response measured for the wind sensor was fitted to that of a Foster network of 20 parallel RC segments (see Fig. A.1). Here, the wind sensor model has been adopted and its RC values have been modified to map its phase, amplitude

**Fig. A.1** The Foster equivalent RC network of the ETF that is used in the time-domain simulations



and step response to that of the ETF, as predicted by the thermal-impedance model (see Fig. 3.16). The resulting  $R$  and  $C$  values are reported in Table 3.2.

The Foster network shown in Fig. A.1 is excited by the square-wave current sources with amplitude  $P$  (ETF heater power). This current (power dissipation) results in an AC voltage (AC temperature variations) at the output of the network (thermopile output). This output is denoted by  $\Delta T$ , and is the difference between the voltages (temperatures)  $T_1$  and  $T_2$ , which represent the *hot* and *cold* junction temperatures of the thermopile, respectively:

$$\Delta T = T_1 - T_2. \quad (\text{A.1})$$

Using super positioning,  $\Delta T$  can be written as a function of each RC segment's voltage (temperature):

$$\Delta T = \sum_{i=1}^{17} T_{RC-i} - \sum_{j=18}^{20} T_{RC-j}. \quad (\text{A.2})$$

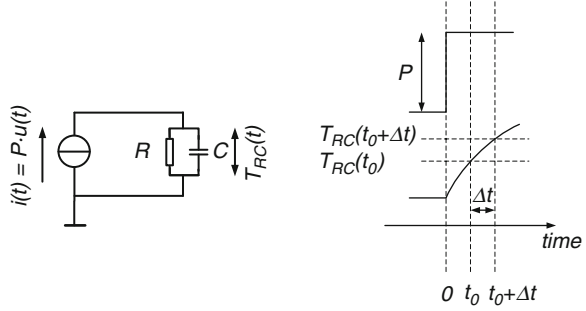
The goal is to calculate the  $\Delta T$  (thermopile output) in the time-domain using the  $P$ ,  $R_i$  and  $C_i$  values, and the heater excitation frequency. Simulations were carried out in Matlab (A.3), where numbers are mainly stored and processed in vectors with an integer number of elements. This means that the time-domain simulations can be done at a limited number of instances, which introduces quantization in time. As a result, a minimum time-step of  $\Delta t$  can be defined as the time space between every two calculation instances of ETF output.

In order to calculate  $T_{RC-i}$  for each segment in (A.2), the step response of a single RC element (shown in Fig. A.2) is considered. The voltage (temperature) across this parallel RC segment, denoted by  $T_{RC}(t)$ , when excited by a step current source  $i(t) = P \cdot u(t)$  is:

$$T_{RC}(t) = P \cdot R \cdot (1 - e^{-t/\tau}). \quad (\text{A.3})$$

where  $\tau = R \cdot C$ .

**Fig. A.2** Discrete-time values of an RC segment step response



As shown in Fig. A.2,  $T_{RC}(t)$  can be calculated at discrete time instances, each  $\Delta t$  seconds apart. The goal is to derive a *generic* equation for  $T_{RC}(t)$  that can be plugged into a *generic* Matlab code for calculation of ETF output signal at discrete time instances. To do so, parameter  $t$  can be expanded into a discrete equation:

$$t = t_0 + n \cdot \Delta t. \tag{A.4}$$

with  $t_0$  as an initial moment of time and  $n$  as an integer. In the next step,  $T_{RC}(t)$  should be calculated at two consecutive instances:  $t_0$  and  $t_0 + \Delta t$  (see Fig. A.2). This allows  $T_{RC}(t_0 + \Delta t)$  to be derived as a function of  $T_{RC}(t_0)$ , which enables the compilation of a *generic* discrete-time equation for  $T_{RC}(n)$ . Using (A.3):

$$T_{RC}(t_0 + \Delta t) = P \cdot R \cdot \left(1 - e^{-\frac{(t_0 + \Delta t)}{\tau}}\right). \tag{A.5}$$

which can be modified by adding and subtracting an extra term:

$$T_{RC}(t_0 + \Delta t) = P \cdot R \cdot \left(1 - e^{-\frac{(t_0 + \Delta t)}{\tau}}\right) + P \cdot R \cdot e^{-\frac{\Delta t}{\tau}} - P \cdot R \cdot e^{-\frac{\Delta t}{\tau}}. \tag{A.6}$$

By re-arranging:

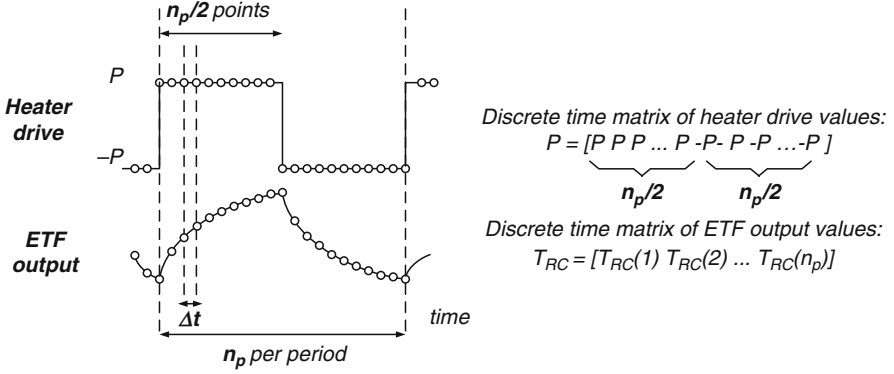
$$T_{RC}(t_0 + \Delta t) = P \cdot R \cdot \left(1 - e^{-\frac{t_0}{\tau}}\right) \cdot e^{-\frac{\Delta t}{\tau}} + P \cdot R \cdot \left(1 - e^{-\frac{\Delta t}{\tau}}\right). \tag{A.7}$$

The first term on the right hand side of (A.7) is equal to  $T_{RC}(t_0)$  and thus:

$$T_{RC}(t_0 + \Delta t) = T_{RC}(t_0) \cdot e^{-\frac{\Delta t}{\tau}} + P \cdot R \cdot \left(1 - e^{-\frac{\Delta t}{\tau}}\right). \tag{A.8}$$

Using (A.4), (A.8) can be expanded into an equation for discrete samples  $n = 1, 2, 3, \dots$  such that:

$$T_{RC}(n) = T_{RC}(n - 1) \cdot e^{-xv} + P \cdot R \cdot (1 - e^{-xv}). \tag{A.9}$$



**Fig. A.3** An illustration of the ETF continuous and discrete-time input and output waveforms, and matrixes that could be programmed in Matlab to hold the discrete-time values

where

$$exv = e^{\frac{-\Delta t}{\tau}}. \quad (\text{A.10})$$

Within a complete period of the ETF heater drive defined by  $T_{\text{drive}} = 1/f_{\text{drive}}$ , a total number of  $n_p$  discrete time instances can be considered (see Fig. A.3). With this assumption, the value of  $\Delta t$  can be calculated as:

$$\Delta t = \frac{T_{\text{drive}}}{n_p}. \quad (\text{A.11})$$

Furthermore, the heater drive signal  $P(n)$  can be written as:

$$P(n) = \begin{cases} P & \text{for } 1 \leq n < \frac{n_p}{2} \\ -P & \text{for } \frac{n_p}{2} \leq n < n_p \end{cases}. \quad (\text{A.12})$$

where  $+P$  and  $-P$  result in a differential thermopile output signal (see Figs. 3.7 and 3.8 for fully differential thermopile structures).

Using (A.2), (A.9), and (A.10), a generic equation can be developed, which calculates the ETF output signal within a period of the heater drive:

$$\underbrace{\Delta T(n)}_{n=1,2,\dots,n_p} = \sum_{i=1}^{17} T_{RC-i}(n-1) \cdot exv_i + P(n) \cdot R_i \cdot (1 - exv_i) - \sum_{j=18}^{20} T_{RC-j}(n-1) \cdot exv_j + P(n) \cdot R_j \cdot (1 - exv_j). \quad (\text{A.13})$$

where

$$exv_k = \frac{-\left(\frac{T_{drive}}{n_p}\right)}{R_k \cdot C_k}. \quad (\text{A.14})$$

The RC values reported in Table 3.2 for the ETF model can be loaded into matrixes to be used in a *generic* Matlab code that calculates the ETF output for a period of the heater drive:

$$\begin{aligned} R &= [R_1 \ R_2 \ R_3 \ \dots \ R_{17} \ R_{18} \ R_{19} \ R_{20}] \\ C &= [C_1 \ C_2 \ C_3 \ \dots \ C_{17} \ C_{18} \ C_{19} \ C_{20}]. \end{aligned} \quad (\text{A.15})$$

The result is the following Matlab code that can be extended to any number of periods by extending the vectors that define the heater drive input:

```

tau = R.*C; % time constant vector
np = 100; % number of points per heater drive period
fdrive = 100e3; % heater drive frequency = 100kHz in this
case
Tdrive = 1/fdrive; % heater drive period
delta_t = Tdrive/np; % time step
power = 2.5e-3; % heater power = 2.5mW in this case
P(1:np/2) = power; % first half period of heater drive
P(np/2+1:np) = -power; % second half period of heater drive
s_tp = 24*0.5e-3; % number of thermocouples times Seebeck
coefficient of one
pol(1:17)=1; % summation polarity for hot junction side
of network
pol(18:20)=-1; % summation polarity for cold junction
side of network
exv = exp(-delta_t./tau); % realization of exv_k in equation A.14
dT(1:20,1) = 0;
for i=2:length(P); % This loop runs per calculation point of
the ETF output
    dT(:,i) = dT(:,i-1)).*exv +
    P(i)*R.*(1-exv); % realization of sigma functions in A.13
    V ETF(i) = pol*
    dT(:,i)*s_tp; % translation to voltage and summation
end; % At the end, the ETF output signal is
stored in V ETF matrix

```

Simulation results for the ETF output signal based on this code, and within a few periods of the heater drive at 100 kHz, and at two levels of the heater power are shown in Fig. 3.35. This code will be embedded into another code that builds the complete time-domain model of the DAFL. This will be discussed in the following section.

## A.2 Time-Domain Simulation of the DAFL

The DAFL system architecture was described in Chap. 4, as depicted by the block-diagram of the loop shown in Fig. 4.25. This loop includes a phase-domain  $\Delta\Sigma$  modulator (PD $\Delta\Sigma$ ), a 12-bit digitally-controlled oscillator (DCO), and a digital integrator. The block-diagram that formed the basis on which the Matlab model of the loop is developed and shown in Fig. A.4.

The output of the RC network representing ETF is fed to the PD $\Delta\Sigma$ , where it is multiplied by the output of a phase DAC. The result is integrated by the continuous-time integrator of the modulator,  $V_{integ\_PDSD}$ , which drives the quantizer. The PD $\Delta\Sigma$ , the DAFL's digital integrator, and the DAC driving the VCO, are sampled at frequency  $f_s$ . The DAC has a total number of bits =  $N_{DAC}$  and a reference voltage =  $V_{ref}$ . The VCO has a voltage-to-frequency gain =  $K_{VCO}$ . The output frequency of the VCO updates a parameter in the loop:  $f_{VCO}$ , from which  $f_{drive}$  of ETF and  $f_s$ , are extracted.

Matlab is a discrete-time environment in which the values of various signals in the loop can be processed and stored in vectors. The loop is simulated for a limited number of sampling periods =  $N_{sample}$ . Each sampling period includes an integer number of ETF heater drive periods =  $2 \times N_{fs}$ , where  $N_{fs}$  represents the number of half-periods. Per half-period of ETF drive signal, there are a limited number of time instances =  $N_{points}$ . The continuous analog signals within the loop, e.g. the ETF output,  $V_{ETF}$ , and  $V_{integ\_PDSD}$ , are only calculated within these time instances. Therefore, the time interval among each two calculation points, denoted by  $\Delta t$ , is the smallest time interval in the whole calculations. In every simulation run of DAFL, the total number of calculation points is equal to:

$$N_{total} = \underbrace{N_{sample}}_{\text{Total number of samples}} \times \underbrace{2 \times N_{fs}}_{\text{Total number of ETF drive periods per loop sample}} \times \underbrace{N_{points}}_{\text{Total number of time instances in one half period of ETF drive}} \quad (\text{A.16})$$

The minimum time interval  $\Delta t$  of the loop is updated every time the DCO output frequency is updated. The value of  $\Delta t$  remains the same (for the calculation of the intermediate points) until the next sampling moment of the loop, when the DCO frequency is updated. Considering a division by 16 between  $f_{VCO}$  and  $f_{drive}$  (see Fig. A.4), the value of  $\Delta t$  can be calculated from:

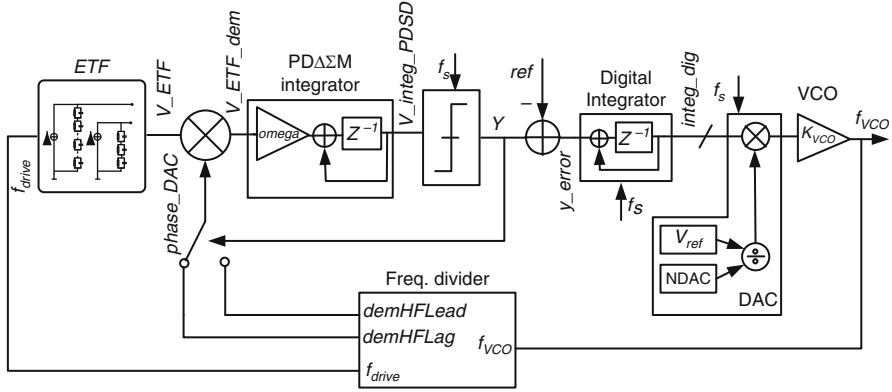


Fig. A.4 Block-diagram of DAFL used for Matlab simulations

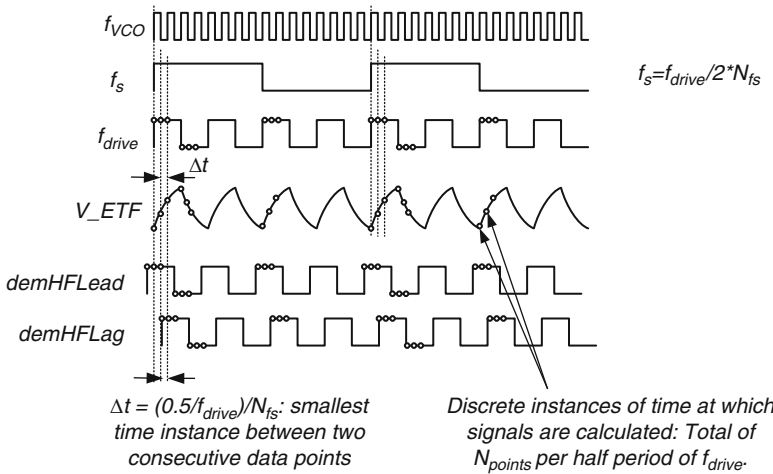


Fig. A.5 Some of discrete-time instances within DAFL continuous-time signals, and the time interval  $\Delta t$  between them

$$\Delta t = 1 / (16 \cdot f_{VCO} \cdot 2 \cdot N_{\text{points}}). \tag{A.17}$$

All the periodic signals within the loop can be defined in matrixes with an integer number of elements. For instance:  $f_{\text{drive}}$ , the signal that drives the ETF heater, can be defined by a vector with an  $N_{\text{total}}$  number of elements [see (A.16)]. In this vector, the value of elements toggle periodically between  $+P$  and  $-P$  ( $P$  is the heater power) at every  $N_{\text{points}}$  element. The phase DAC reference signals can also be defined in the same way. These are equivalent to the  $f_{\text{drive}}$  vector, with the difference being that their elements toggle between  $+1$  and  $-1$ . The lead and lag phase shifts of the phase DAC references, in reference to  $f_{\text{drive}}$ , can be simply implemented by shifting the vector elements forwards and backwards (see Fig. A.5) by a total number of:

$$N_{shift} = 2 \cdot N_{points} \cdot \frac{phase\_ref}{360^\circ}. \quad (\text{A.18})$$

where  $phase\_ref$  can for instance be equal to  $\pm 45^\circ$ .

Once the signals in the DAFL system block diagram are pre-defined, an algorithm (see Fig. A.6) can be defined that runs for the total number of samples and calculates the intermediate points for the internal signals. This algorithm involves three loops. The *most inner loop* runs per half-period of the ETF heater drive signal. This is a *for loop* in Matlab that runs for  $N_{points}$ . Within this loop, the ETF output signal  $V_{ETF}(n)$ , and the result of its multiplication with the phase DAC output,  $V_{ETF\_dem}(n)$ , are calculated. Furthermore, this loop calculates the PD $\Delta\Sigma$ 's integrator output,  $V\_integ\_PDS$ . Consequently, the integrator's continuous-time output signal is calculated every time one of the intermediate ETF output signal points is calculated. This reproduces the continuous-time integrator's signal in an over-sampled fashion in regard to the modulator's sampling frequency.

An ideal continuous-time integrator has a frequency-domain transfer:

$$H(s) = \frac{\omega_0}{s}. \quad (\text{A.19})$$

in which  $\omega_0$  is its unity-gain frequency (Fig. A.7a), i.e. the frequency at which the amplitude response of the integrator reaches the 0 dB gain point. In the time-domain, the output of such integrator will be a ramp in response to a unit step input (Fig. A.7b). After  $\Delta t$  seconds the integrator output growth (from zero initial condition) is:

$$\Delta V_{out}(\Delta t) = \omega_0 \cdot \Delta t. \quad (\text{A.20})$$

In the discrete time simulation environment of Matlab, a calculation of the continuous-time integrator output at the  $i$ th discrete sample can be written as:

$$V_{out}(i) = V_{out}(i-1) + v_{in}(i) \cdot \omega_0 \cdot \Delta t. \quad (\text{A.21})$$

For the DAFL,  $\Delta t$  can be calculated from (A.17).

The *intermediate loop* of the DAFL algorithm (see Fig. A.6) runs for the  $f_{drive}$  half periods within one sampling period of the loop. Therefore, the loop runs for  $2 \cdot N_{fs}$  times between every two samples. The *most outer loop* runs for the total number of samples  $N_{sample}$ . Within this loop, the PD $\Delta\Sigma$  quantizer and the phase DAC outputs are updated. Also within the same loop, the bitstream of the modulator,  $Y$  (see Fig. A.4), is compared with the DAFL phase reference,  $ref$ . This is a consecutive set of one's and zero's at a rate of  $f_s/2$  and represents a  $90^\circ$  phase shift in the ETF (for  $phase\_ref$  values of  $\pm 45$ ). The result of this comparison is integrated by the digital integrator, whose output then updates the DAC input



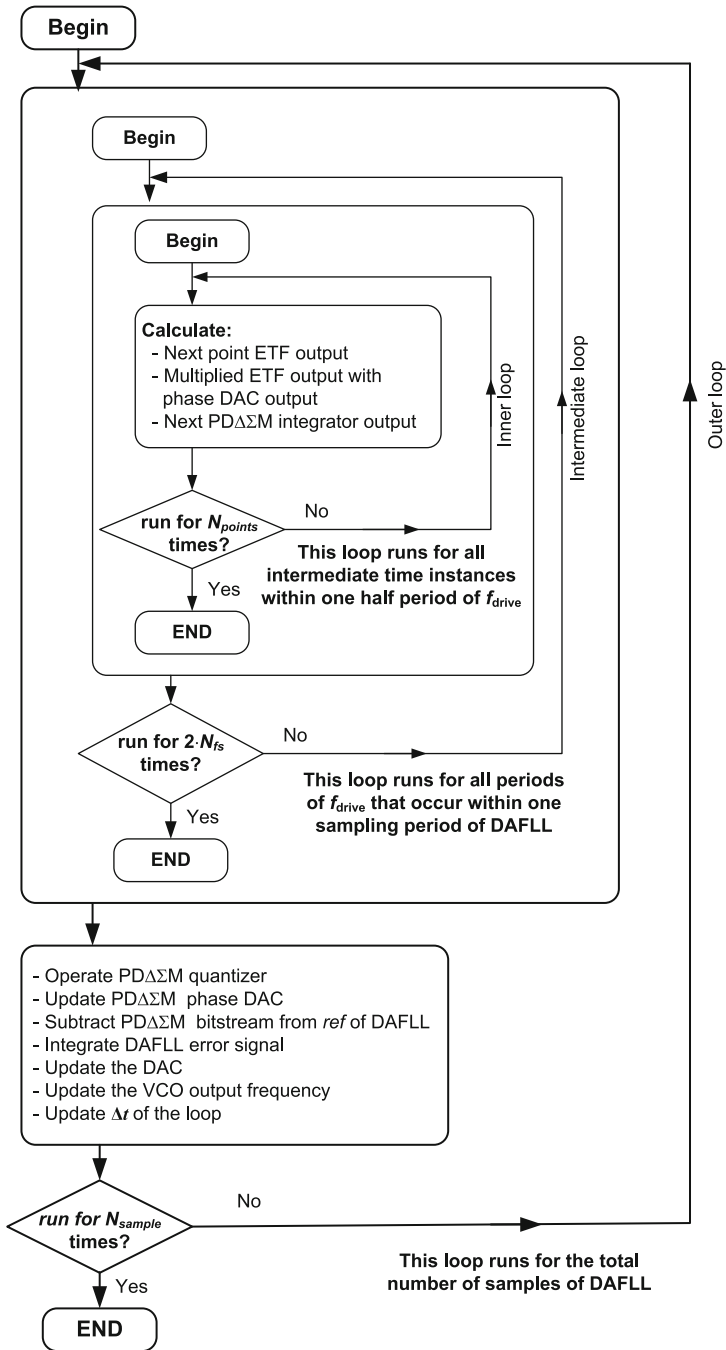
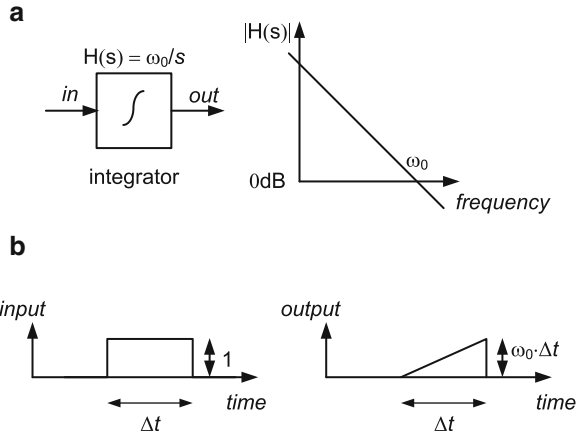


Fig. A.6 The DAFL time-domain calculation algorithm

**Fig. A.7** A continuous-time integrator: (a) frequency-domain response; (b) step-response



every time the outer loop runs. This then changes the VCO output frequency,  $f_{VCO}$ , which updates  $f_{drive}$  and  $\Delta t$  eventually.

Based on the proposed algorithm a Matlab code can be devised which forms the basis for the time-domain simulations of the DAFL shown in Fig. A.4. This code can be further expanded for the addition of non-idealities or the study of loop dynamics in the time domain. A few simulation results, such as the loop startup and step response, as well as the output spectrum of the PD $\Delta$  $\Sigma$ M and digital integrator, were shown in Chap. 4 and in Figs. 4.26 and 4.27. The generic Matlab code that implements the proposed algorithm is as follows:

```

% Initial definitions
s_tp = 24*0.5e-3;           % Thermopile sensitivity
kvco = 0.9e6;              % VCO sensitivity [Hz/V]
Nsample = 8192;           % Number of samples of the loop
Nfs = 2;                  % Number of ETF drive periods per sample
Nperiods = 2*Nfs*Nsample; % Number of half periods of ETF drive to be
                           % simulated
Npoints = 24;            % Number of intermediate points in every
                           % half period of fdrive
f = 100e3; T = 1/f;      % Initial condition for ETF drive frequency
fvco = f*ones(1, Nsample); % Define a vector that holds fVCO values
vco_in = zeros(1, Nsample); % Define a vector holding VCO input signal
fs = f/Nfs;              % Sampling frequency of DAFL
fT = 300e3;              % 0-dB frequency of the continuous-time
                           % integrator [Hz]
NDAC=12; Vref=4;        % DAC definition

```

```

demHFLead = zeros (1, Nperiods*Npoints); % phase DAC's Lead feedback
                                             signal
demHFLog = zeros (1, Nperiods*Npoints); % phase DAC's Lag feedback
                                             signal
V_integ_PDS =
  zeros(1, Nperiods*Npoints); % PDS's integrator signal
y = ones(1, Nsample); % PDS's bitstream

% DAFLL reference bit stream (90 degrees phase), and error signal of
the loop
yerror = zeros(1, Nsample);
diginteginit = 256; % The initial condition at which the digi-
                    tal integrator starts

integ_dig = diginteginit*
  ones(1, Nsample); % error signal pre-allocation
ref = zeros(1, Nsample); % DAFLL reference signal
ref(3:2:Nsample)=1; % DAFLL reference signal

% Definition of the ETF drive square-wave
range = (1:2*Npoints: Nperiods*Npoints); % initialization of
                                             vectors
fdrive = zeros (1, Nperiods*Npoints); % initialization of vectors
fdrive(range) = 1; % initialization of vectors
cycle = [ones(1, Npoints),
  zeros(1, Npoints)]; % initialization of vectors
fdrive = filter
  (cycle, 1, fdrive); % initialization of vectors
fdrive = 2*fdrive - 1;
demHFLead = circshift (fdrive, [1 -Npoints/4]); % Definition of the
                                             Lead feedback
demHFLog = circshift (fdrive, [1 Npoints/4]); % Definition of the
                                             lag feedback
phase_DAC = demHFLead; % Initial phase DAC
                                             value

P = 25/(2*1.2e3); % The heater drive power [W]
R = etf_model(:, 2); % The vector including ETF's Foster net-
                    work resistances
C = etf_model(:, 3); % The vector including ETF's Foster net-
                    work capacitances
pol(1:17)=1; % polarity of summation for hot junction
             side of network
pol(18:20)=-1; % polarity of summation for cold junction
              side of network
tau = R.*C; % Time constant vector
V ETF = zeros (1, Nperiods*Npoints); % pre allocating thermopile
                                         output signal
V ETF_dem = zeros (1, Nperiods*Npoints); % pre allocating demodulated
                                         ETF output

ix = 1;
% Assignment of startup values so that the loop can start working
delta_t = T/(2*Npoints);

```

```

exv = exp(-delta_t./tau);
dT(1:20,1) = 0;
% An initial condition for the continuous time integrator's cut off
frequency
omega = 2*pi*fT/fs/(2*Npoints*Nfs);
% The DAFL loop simulation
% The outer loop runs per loop sample
for sdx = 2:Nsample
% The mid loop runs per half period of fdrive in one sampling period
of sigma-delta
    for jx = 2:2*Nfs+1
        % The inner loop runs per intermediate calculation steps
        for px = 1:Npoints
            % Calculating the ETF output value for the current intermediate
            point
            ix = ix + 1;
            ix = min(Npoints*2*Nfs*Nsample,ix);
            dT(:,ix) = dT(:,(ix-1)) .*exv + P*fdrive(ix)*R.*(1-exv);
            V ETF(ix) = pol*dT(:,ix)*s_tp;
            % multiply ETF output with phase DAC feedback phase
            V ETF_dem(ix) = V ETF(ix)*phase_DAC(ix);
            % phase-domain delta sigma's integrator value evaluation
            V_integ_PSD(ix) = V_integ_PSD(ix-1) + omega*V ETF_dem(ix);
        end
    end
end
% Quantizer
y(sdx) = sign(V_integ_PSD(ix));
% Phase DAC
if (y(sdx)==1)
    phase_DAC = demHFLead    else
    phase_DAC = demHFLag    end;
% DAFL phase summation node: subtraction, error signal generation
yerror(sdx) = ref(sdx) - ((y(sdx)*-1)+1)/2;
%subtract PSD bitstream & phase ref
integ_dig(sdx) = integ_dig(sdx-1) +yerror(sdx-1); %digital integrator

% Update the DCO DAC value
vco_in(sdx) = 4*(integ_dig(sdx))*(Vref/2^(NDAC-1));
% Update the VCO frequency
fvco(sdx) = max(kvco*vco_in(sdx),100e3);
% Feedback the VCO output to the ETF, i.e. close the DAFL loop
feedback
f = fvco(sdx)/16;
T = 1/f;
fs = f/Nfs;

```

```
delta_t = T/(2*Npoints); % update the time stamp of the loop based
on latest fVCO
exv = exp(-delta_t./tau);
% Update the PSD's continus time integrator cut off frequency
omega1 = 2*pi*fT/fs/(2*Npoints*Nfs);
end
```

# Summary

This book investigates the concept of generating accurate on-chip frequencies based on the thermal properties of silicon. The work presented here shows that electrothermal (thermal-diffusivity-based) frequency references are feasible in standard CMOS processes. Furthermore, the possibility of scaling such references into the more modern CMOS processes is studied. The improvements achieved in the performance of the scaled reference show that such frequency references strongly benefit from process scaling.

A frequency reference is an indispensable part of most electronic systems. In recent years, a lot of effort has been devoted to the replacement of quartz crystal oscillators with integrated solutions. The main motivations have been the reduction in size and cost as well as the increase in reliability of electronic circuits. Chapter 2 provides an overview of the recent developments in the implementation of silicon-based frequency references. In this survey, various types of oscillators including MEMS, RC, LC, ring, and electron-mobility-based are studied. The MEMS and LC oscillators have been commercialized, so far. These references achieve stabilities of  $< 1$  ppm and 50 ppm, respectively, enabling them to compete with the quartz crystal oscillators.

Apart from some earlier thermal oscillators, the thermal properties of silicon have received much less attention in generation of on-chip frequencies. These concepts are discussed in Chap. 3. In principle, the heat generated in a silicon substrate diffuses at a defined rate which is determined by the thermal diffusivity of silicon,  $D$ . This forms the basis for the thermal delay lines, around which the thermal oscillators are made. A thermal delay line embeds a heater at a distance  $s$  from a temperature sensor within a silicon substrate. Its delay is then governed by  $D$  and  $s$ . The former is a stable property of IC-grade silicon, while the latter is defined by lithography. The early thermal oscillators had large levels of output jitter and high levels of power consumption. These were mainly due to heat losses in the substrate, which resulted in very small signals at the output of their temperature sensors.

Chapter 3 further describes how a thermal delay line can be extended into a more complex structure called an electrothermal filter (ETF). An ETF realizes a heater

(e.g. a resistor) at a close distance  $s$  (less than  $25\ \mu\text{m}$ ) from a relative temperature sensor (e.g. a thermopile), and thus can be realized in standard CMOS processes. This structure behaves like a low-pass filter, whose well-defined phase response is determined by  $D$  and its geometry. Measurements on ETFs have shown tolerances in the order of 0.1%. Chapter 3 describes how embedding an ETF into a feedback loop with a narrow noise-bandwidth solves the unacceptable jitter performance of the previous thermal oscillators. Such loop is called an electrothermal frequency-locked loop (FLL). An FLL locks the output frequency of a VCO to the accurate phase shift of the ETF. As a result, this frequency is determined only by the ETF and thus not affected by the VCO's tolerances and drifts. Earlier implementations of electrothermal FLLs are investigated in Chap. 3. Since the output frequency of an FLL follows the same  $T^{-1.8}$  temperature dependence of  $D$ , these FLLs were aimed as accurate temperature-to-frequency converters.

The temperature-dependent output frequency of FLLs showed promising levels of untrimmed device-to-device frequency spread, which was in the order of  $\pm 0.25\%$  over the industrial temperature range. Chapter 3 concludes that such levels of inaccuracy imply that a temperature-compensated electrothermal FLL should form a suitable basis for the realization of accurate electrothermal frequency references.

There were challenges associated with the integration of early FLLs. These required large off-chip capacitors in order to implement their narrow noise-bandwidths, making their CMOS integration rather difficult. Therefore, Chap. 4 proposes a digitally-assisted electrothermal FLL (DAFLL), in which the narrow noise-bandwidth of the loop is implemented by a digital filter. A DAFLL incorporates a digital phase detector (a phase-domain  $\Delta\Sigma$  modulator) that digitizes the phase shift of an ETF, a digital phase reference, a phase summation node, and a digitally-controlled oscillator (DCO). An implementation in a standard  $0.7\ \mu\text{m}$  CMOS achieved an output frequency with an untrimmed device-to-device spread of  $\pm 0.3\%$  ( $3\sigma$ ) from  $-55^\circ\text{C}$  to  $125^\circ\text{C}$ . This level of inaccuracy is comparable to the early FLLs. Furthermore, the output frequency of the loop followed the same  $T^{-1.8}$  temperature dependence, associated with the thermal diffusivity of silicon. The DAFLL provided a solution for the integration challenge of electrothermal FLLs and therefore was adopted as the foundation of the electrothermal frequency references proposed in this book.

Chapter 5 describes how an electrothermal frequency reference can be realized by temperature-compensating a DAFLL. This was done by measuring the temperature of the die using an on-chip band-gap temperature sensor and injecting the digital temperature information into the loop. The result was the first fully-integrated electrothermal (thermal-diffusivity-based) frequency reference, realized in a standard  $0.7\ \mu\text{m}$  CMOS process. The reference has an output frequency of 1.6 MHz, dissipates 7.8 mW, and achieves an absolute inaccuracy of  $\pm 0.1\%$  over the military temperature range ( $-55^\circ\text{C}$  to  $125^\circ\text{C}$ ) with a single room-temperature trim. Its worst-case temperature coefficient of  $\pm 11.2\ \text{ppm}/^\circ\text{C}$  allows for trimming without temperature stabilization, which simplifies the trimming procedure and thus reduces trimming costs.

Chapter 6 describes the scaling of an electrothermal frequency reference into a more modern CMOS process. The performance of an electrothermal frequency reference is mainly determined by its ETF. The accuracy and phase-frequency characteristic of an ETF are functions of its geometry. Scaling the ETF enables trade-offs regarding the accuracy, power consumption, output frequency and jitter. By scaling the ETF and adopting a more modern CMOS technology, a scaled electrothermal frequency reference was developed in a standard 0.16  $\mu\text{m}$  CMOS process. The ETF's critical dimension  $s$  (the distance between heater and temperature sensor) was scaled from 24 to 4.7  $\mu\text{m}$ , which results in  $5.5\times$  more SNR and allows for a  $10\times$  higher frequency. The improved lithographic accuracy of the 0.16 $\mu\text{m}$  process (compared to the previous 0.7  $\mu\text{m}$  process) maintains its accuracy. The resulting frequency reference has an output frequency of 16 MHz ( $10\times$  higher compared to previous work), occupies an area of 0.5  $\text{mm}^2$  ( $12\times$  smaller), dissipates 2.1 mW ( $3.7\times$  less), and has an rms output jitter of 45 ps ( $7\times$  less). Measurements on 24 devices show that the reference maintains the  $\pm 0.1\%$  level of inaccuracy (from  $-55^\circ\text{C}$  to  $125^\circ\text{C}$ ) achieved by the previous work with a single room-temperature trim.



## About the Authors

**S. Mahdi Kashmiri** was born on March 21st, 1980 in Tehran, Iran. He received his B.Sc. degree in electrical engineering from Tehran University, Tehran, Iran, in 2001, and his M.Sc. degree in microelectronics (*cum laude*) from Delft University of Technology, Delft, The Netherlands in 2006. In April 2012, he received his Ph.D. degree from the same university for his work on electrothermal frequency references in standard CMOS. From June 2001 to August 2004, he was a system design engineer at Parman Co. Tehran, Iran, where he worked on the development of a SDH fiber optics telecommunication system. From September 2005 to October 2006, he was an intern at the mixed-signal circuit and systems group of Philips Research Laboratories (currently NXP research), Eindhoven, The Netherlands, where he worked on a wide-band continuous-time sigma-delta modulator. Since October 2010 he has been with the precision systems group of Texas Instruments Incorporated, Delft, The Netherlands (formerly National Semiconductor Corporation). His research interests include the analog and mixed-signal integrated circuits, precision analog systems, and data converters.

Mr. Kashmiri received the 2009 Young Scientist Award of the European Solid-State Circuits Conference (ESSCIRC).

**Kofi A.A. Makinwa** received the B.Sc. and M.Sc. degrees from Obafemi Awolowo University, Nigeria in 1985 and 1988 respectively. In 1989, he received the M.E.E. degree from the Philips International Institute, The Netherlands and in 2004, the Ph.D. degree from Delft University of Technology, The Netherlands.

From 1989 to 1999, he was a Research Scientist with Philips Research Laboratories, Eindhoven, The Netherlands, where he worked on interactive displays and on front-ends for optical and magnetic recording systems. In 1999, he joined Delft University of Technology, where he is now an Antoni van Leeuwenhoek Professor in the Faculty of Electrical Engineering, Computer Science and Mathematics. His main research interests are in the design of precision analog circuitry, sigma-delta modulators, smart sensors and sensor interfaces. This has resulted in 4 books, 18 patents and over 150 technical papers.

Kofi Makinwa is on the program committees of the European Solid-State Circuits Conference (ESSCIRC) and the Advances in Analog Circuit Design (AACD) workshop. He has also served on the program committee of the International Solid-State Circuits Conference (ISSCC), as a guest editor of the Journal of Solid-State Circuits (JSSC) and as a distinguished lecturer of the IEEE Solid-State Circuits Society (2008–2011). He is a co-recipient of several best paper awards: from the JSSC, ISSCC, Transducers and ESSCIRC, among others. In 2005, he received a Veni Award from the Netherlands Organization for Scientific Research and the Simon Stevin Gezel Award from the Dutch Technology Foundation. He is an alumnus of the Young Academy of the Royal Netherlands Academy of Arts and Sciences and an elected member of the IEEE Solid-State Circuits Society AdCom, the society's governing board.

# Index

## A

AC, 7, 47–49, 53, 56, 65, 88, 93, 96, 99–102  
Analog-to-digital converter (ADC), 10, 38, 89,  
135, 138, 171, 172  
Auto-zeroing, 27, 164, 165, 174

## B

Band-gap, 10, 11, 26, 27, 31, 32, 34, 36, 38,  
130, 134, 137–146, 148, 150, 154, 159,  
161, 171, 187, 188  
Band-gap temperature sensor, 10, 11, 36, 38,  
130, 134, 137–145, 159, 171, 172,  
187, 188  
Bar electrothermal filter, 58, 59, 66, 92, 98,  
107–111  
Base-emitter voltage, 32, 137, 171, 172  
Batch calibration, 179  
Bias circuit, 32, 98, 105–107, 140, 141,  
172, 173  
Binary, 2, 112, 117, 171  
Binary-to-thermometer, 171  
Bipolar transistor, 10, 32, 52, 130, 134, 137,  
141, 143, 145, 172  
Bitstream, 89, 90, 93, 95, 97, 108, 134, 135,  
138, 141, 143, 144, 160, 163, 173, 175

## C

Calibration, 148, 179–181  
Capacitance, 18, 21, 30, 31, 35, 37, 47, 56, 60,  
61, 68, 84, 99, 100, 102, 104, 105, 107,  
122, 145, 163, 165, 167–169, 178, 187  
Capacitor, 10, 21, 25–29, 31–33, 35, 37, 66–68,  
84, 88, 94, 98, 100, 104, 115, 124,  
141–145, 169, 170, 172–175

Cascode, 24, 25, 98, 101–105, 115, 118, 163,  
166, 167, 173, 187  
Charge balancing, 138–141, 144, 172–174  
Charge injection, 144, 161, 175  
Chopper, 65–68, 79, 94, 98, 99, 101–104, 108,  
161, 163, 164, 167–170, 175, 187  
Closed-loop, 32–35, 76, 79–81  
Closed-loop compensation, 32–35  
CMFB. *See* Common-mode feedback (CMFB)  
CMOS, 1, 5, 6, 8, 9, 11, 12, 16–20, 22, 23,  
25–27, 29–32, 34, 36, 38–41, 45, 46,  
51–54, 58, 63, 84, 85, 87–125, 129–150,  
153–182, 187–189  
Cold junction, 52–58, 156–158  
Common-mode feedback (CMFB), 98, 99, 167,  
169, 173  
Comparator, 26–29, 31, 34, 35, 37, 38, 46, 78,  
115–117, 164, 169–171  
Complementary-to-absolute temperature  
voltage (CTAT), 137, 171, 172  
Current source, 36, 37, 48, 98, 99, 101, 102,  
117, 118, 141, 173  
Current-steering, 117–118, 135  
Cycle-to-cycle jitter, 4, 29, 83, 153, 171,  
180, 183

## D

DAC. *See* Digital-to-analog converter (DAC)  
DAFLL. *See* Digitally-assisted  
FLL (DAFLL)  
DCO. *See* Digitally-controlled oscillator  
(DCO)  
Dead band, 100, 103, 166  
Decoder, 112, 171  
DEM. *See* Dynamic element matching (DEM)

- Differential, 24, 30–32, 53, 54, 63, 64, 94, 98, 99, 104–106, 116, 118, 156, 169
- Diffusion, 7, 49, 51, 52, 56, 79, 122, 155
- Digital filter, 89, 90, 123, 125, 146
- Digitally-assisted frequency-locked loop (DAFLL), 10, 11, 87–88, 92, 96, 98, 110–115, 118, 120–125, 129–136, 145, 146, 150, 154, 155, 161, 171, 187–189
- Digitally-controlled oscillator (DCO), 11, 87, 89–92, 111–115, 118, 119, 129, 134–136, 146, 150, 154, 159–161, 169–171, 178, 181, 189
- Digital-to-analog converter (DAC), 10, 83, 89, 93, 112, 114–119, 123, 135, 160, 163, 169–171, 178, 179, 189
- Dynamic element matching (DEM), 141, 144
- Dynamics, 11, 45, 70–75, 141, 173
- E**
- Effective thermal diffusivity, 120–125
- Electrothermal filter (ETF), 7–11, 45–48, 51–63, 65–76, 78–85, 87–98, 100, 101, 103, 107–111, 113, 118–125, 129–132, 134–136, 146, 148–150, 153–169, 176–178, 180, 188, 189
- design, 54–60, 80
- simulation, 109
- Electrothermal frequency-locked loop, 8, 10, 11, 45, 66, 68–76, 78, 79, 83, 84, 87, 90, 118, 119, 123–125, 131
- Electrothermal frequency reference, 1, 9–12, 15, 41, 45, 46, 64, 129–150, 153–184, 188
- Error budget, 90, 137, 178
- ETF. *See* Electrothermal filter (ETF)
- F**
- Feedback, 8, 23, 24, 28, 29, 31, 33–35, 39, 46, 62, 63, 65, 74, 75, 78–81, 83, 84, 89, 95, 96, 98, 102, 103, 107, 116, 123, 129, 131, 138, 140, 154, 165, 169, 172, 173
- Feed-forward, 139–141, 173, 175
- Frequency-locked loop (FLL), 8–11, 34, 35, 45, 64–85, 87–125, 129, 131–134, 146, 148, 179
- Flicker ( $1/f$ ) noise, 22, 27, 29, 101, 103, 117, 171, 173, 181
- Folded-cascode, 98, 101, 103, 163, 173
- Frequency, 1–12, 15–41, 45–85, 87–125, 129–150, 153–184, 187–189
- Frequency-domain, 49, 112
- Frequency reference, 1–6, 8–12, 15–41, 45–85, 87, 125, 129–150, 153–184, 187–189
- G**
- Gain booster, 24, 118, 163, 167
- Gain boosting, 102–105, 168, 187
- Geometry, 7–10, 17, 45–48, 52, 54, 55, 69, 71, 84, 96, 108, 120, 122, 124, 150, 153, 155
- H**
- Heat, 1, 6–11, 24, 46–51, 57, 59, 62–64, 67, 84, 120–122, 176
- Heater, 6–8, 46–60, 62–66, 69, 71, 73, 75–77, 80–84, 88, 97, 114, 122, 134, 154–156, 158, 161, 176–178
- Heater drive, 8, 64, 67, 71, 72, 88, 114, 161, 164, 165, 176–178
- Hot junction, 52, 53, 56, 58, 60, 156
- Hysteresis, 116, 171
- I**
- Impedance, 24, 25, 27, 48–51, 53, 55–62, 71, 73, 74, 80, 82, 94, 100–103, 158
- Inaccuracy, 2, 3, 8, 10, 24, 27, 31, 36, 51, 67, 68, 90, 91, 94, 100, 110, 120, 121, 130, 132, 134–136, 139, 146, 147, 149, 150, 153, 154, 160, 161, 172, 178–180, 183, 187–188
- Incremental conversion, 173
- Inductance, 21
- Inductor, 1, 5, 20–22, 41
- Integrator, 8, 10, 29, 33, 65, 68, 72, 74–84, 88–90, 93–98, 100, 112, 114, 115, 135, 139, 141, 143, 144, 146, 163–166, 168–169, 173–175
- J**
- Jitter, 4, 8–12, 17, 19–23, 27–30, 32, 39, 40, 45, 62, 64–69, 74, 78–85, 87, 88, 117, 119, 150, 153, 155, 160, 171, 178–181, 183, 186, 187
- Junction, 31, 52–58, 60, 121, 156–158
- K**
- kT/C noise, 144

**L**

Latch, 26–28, 94, 105, 115, 116, 169, 170  
 LC oscillators, 5, 16, 20–23, 38, 40, 41, 112, 181, 186, 187  
 Leakage, 100, 163  
 Lithography, 9, 17, 45, 51, 84, 90, 124, 135, 150, 154  
 Loop filter, 10, 11, 19, 33, 72, 74, 87, 114, 138, 139, 141–143  
 Low-frequency chopper, 67, 108, 109  
 Low-pass filter, 7, 45, 65, 80

**M**

Micro electro mechanical system (MEMS), 1, 5, 11, 16–21, 39, 40  
 Micro electro mechanical system based oscillator, 16–20, 38, 40  
 Mismatch, 103, 139, 141, 144, 161, 168, 171, 178  
 Mobility-based, 11, 36–38, 40  
 Modulator, 11, 89–91, 93–98, 100, 107, 108, 111, 134, 138–144, 146, 159–164, 168, 169, 172–175

**N**

Noise, 3, 4, 8, 10, 11, 15, 17, 19, 22, 27–29, 52, 55–57, 60, 62, 64–66, 68, 69, 72, 78–85, 88, 89, 91, 97, 98, 101, 103, 108, 114, 116, 117, 119, 124–125, 135, 141, 143–146, 154, 155, 157, 160, 163, 164, 168, 171, 173, 175, 178, 181, 189  
 bandwidth, 10, 11, 65, 68, 69, 74, 80–82, 84, 85, 90, 91, 97, 108, 119, 124–125, 135  
 shaping, 89, 114  
 transfer, 79–82

**O**

Offset, 4, 27, 28, 52, 66–68, 94–98, 101–105, 118, 139, 160–165, 167–169, 172, 173, 175, 187  
 Opamp, 28, 68, 140, 163–165, 168–169, 172  
 Open-loop, 31–32, 66–67, 104, 105  
 Optimized electrothermal filter, 58–60, 92, 97, 98, 108, 110, 111, 118, 132, 136

**P**

Parts-per-million (ppm), 3, 5, 10, 15, 16, 18–25, 27, 32, 34, 38–41, 51, 69, 117, 134, 136, 149, 150, 171, 179, 180, 184, 189

PD $\Delta$  $\Sigma$ M, 11, 87, 89–107, 111–115, 119, 123, 134–136, 146, 149, 159–169, 176, 178  
 Phase digital-to-analog converter, 93, 163, 178  
 Phase-domain delta sigma, 89, 108, 111  
 Phase error, 51, 90–91, 94–97, 99, 135, 136, 146, 161–164, 179  
 Phase-frequency characteristic, 87, 124, 131, 132, 155  
 Phase-locked loop (PLL), 19, 20, 30, 74  
 Phase-margin, 99, 104, 167–169, 173  
 Phase noise, 4, 17, 28  
 Phase reference, 74, 89–91, 93, 96, 113–114, 131, 135, 146, 154, 159, 161, 163, 164, 176  
 Phase shift, 8, 10, 24, 25, 50, 51, 55–58, 60, 63–67, 70–72, 74–76, 82, 84, 88–94, 96, 97, 99, 100, 110, 113–114, 122–124, 132, 134, 135, 150, 154, 156, 158–161, 168  
 Polynomial, 19, 38, 135, 146–148, 179, 182  
 Polynomial fit, 108  
 Process, voltage, and temperature (PVT), 3, 28, 31, 32, 34, 35, 40, 136  
 Proportional-to-absolute temperature voltage (PTAT), 23, 137–140, 145, 148, 171–173, 180

**Q**

Quantization noise, 91, 114, 141, 143, 160, 163, 175  
 Quantizer, 93, 94, 138, 144, 164  
 Quartz, 1, 2, 4, 5, 9, 16–20  
 Quartz crystal, 1, 2, 4, 5, 9, 16, 17, 19, 20

**R**

R-2R ladder, 117  
 RC filter, 56, 99  
 RC oscillators, 5, 16, 23–26, 63, 160, 181  
 Relaxation oscillator, 7, 25–29, 36, 38, 61, 62, 78, 112, 114–117, 135, 169, 189  
 Residual offset, 10, 67, 68, 94–98, 101–103, 161–163, 167–169, 175, 187  
 Resistance, 8, 19, 21, 22, 24, 26, 35, 47, 55, 56, 59–61, 79, 80, 89, 122, 155–158, 171, 176  
 Resistor, 1, 5, 7, 22–24, 26, 28, 29, 31, 34, 35, 37, 41, 51, 59, 99, 105, 107, 117, 118, 134, 155, 160, 169, 170  
 Resolution, 19, 27, 38, 55, 89, 91, 98, 108, 110, 112, 120, 136, 144, 145, 150, 159, 161, 163, 172, 178, 179, 189  
 Resonance, 17–20, 22, 23  
 Resonator, 1, 5, 11, 17–20, 38, 40

Ring oscillators, 17, 30–35, 38, 188  
 Ripple, 10, 67, 68, 97, 124, 164, 165, 175

## S

Sampling, 88–91, 100, 108, 114, 119, 141–145, 160, 161, 163, 164, 169, 173, 174  
 Saturation current, 137, 148  
 Scaled, 1, 9, 12, 138, 153–184, 188, 189  
 Scaled electrothermal filter, 155–159, 161, 164, 165, 168, 176  
 Scaled electrothermal frequency reference, 153–184, 188  
 Scaling, 1, 9, 12, 45, 76, 153–159, 161, 173, 184, 188  
 Seebeck, 6, 52, 55, 57, 157  
 Signal-to-noise-ratio (SNR), 55–58, 60, 157–159  
 Silicon-based, 1, 5, 9, 11, 15–41, 188  
 Sinc filter, 108  
 Single capacitor sampling, 144–145, 172  
 Single trim, 36, 39, 149, 180  
 SNR. *See* Signal-to-noise-ratio (SNR)  
 Spectrum, 4, 108, 109, 114, 146  
 Stability, 1–3, 5, 8, 9, 15–17, 19–24, 26, 28, 32, 36, 38, 40, 41, 46, 70, 104, 105, 111, 112, 160, 180  
 Steady-state, 34, 72–75, 80–82, 98  
 Step, 3, 10, 16, 20, 61, 62, 72, 75–78, 87, 91, 111, 112, 114, 119, 159, 171, 172  
 Step response, 61, 62, 75–77, 111, 114  
 Substrate, 1, 6–9, 18, 45, 46, 48–53, 59, 69, 84, 122, 124, 125, 134, 135, 138, 141, 154, 171, 187  
 Synchronous demodulator, 8, 65, 67, 72, 73, 76, 78, 83, 84, 88, 94, 95, 98–100, 164, 176, 178  
 Synthesizer, 18–21  
 System-level, 10, 11, 16, 72, 87, 89–98, 111–114, 132–140, 144, 159–160, 162, 172, 189

## T

TD. *See* Thermal-diffusivity-based (TD)  
 TDC. *See* Temperature-to-digital converter (TDC)  
 Temperature  
   coefficient, 18, 21, 22, 24, 27, 31, 32, 34, 35, 39, 51, 130, 148, 150, 180  
   compensation, 5, 8–11, 16, 18, 19, 22, 23, 26, 31, 32, 36, 38, 41, 69, 70, 87, 91, 125, 129–150, 171, 187, 188  
   dependent/dependence, 3, 7–10, 16, 19, 21, 23, 24, 26, 27, 31, 32, 36, 38, 39, 49, 51,

66, 69, 71, 72, 84, 91, 93, 96, 115, 125, 129–135, 137, 150, 173, 188  
 sensor, 6, 7, 9–12, 18, 19, 36, 38, 46, 48–52, 62–64, 66, 69, 71, 73, 75, 78, 82, 84, 92, 94, 96, 97, 108, 129, 130, 132, 134, 137–145, 148, 154, 159, 161, 171–175, 187, 188  
 Temperature-to-digital converter (TDC), 92, 100, 109, 110  
 Thermal delay, 6, 7, 10, 46, 48–50, 61, 62, 71, 84  
 Thermal diffusivity, 1, 7–9, 11, 45, 49, 66, 69, 71, 75, 84, 85, 87, 91, 92, 120–125, 129, 130, 132, 137, 145, 150, 153, 154, 161, 184, 187–189  
 Thermal-diffusivity-based (TD), 9, 92, 129, 132, 137, 145, 153, 161, 178, 184, 189  
 Thermal-domain, 6  
 Thermal impedance, 48–51, 53, 55–62, 71, 73, 74, 80, 82, 94, 158  
 Thermal noise, 8, 11, 45, 56, 57, 60, 62, 66, 72, 78–85, 87, 89, 91, 97, 98, 101, 108, 119, 155, 157, 163, 173, 189  
 Thermal oscillator, 1, 6–9, 11, 45, 46, 61–65, 69, 84  
 Thermocouple, 6, 51–58, 60, 62, 63, 75, 79, 82, 155–157  
 Thermopile, 7, 8, 51–60, 65–67, 75, 79–81, 89, 96, 99, 100, 114, 122, 154–158, 163, 167, 168, 176  
 Time-domain analysis, 60–61  
 Transconductance, 22, 24, 67, 68, 87, 97, 100, 105, 107, 165, 167  
 Transconductor, 21, 66, 67, 93, 94, 96–103, 105–107, 163–168  
 Trim, 9, 22–24, 27, 36, 39, 139, 143, 146, 148–150, 170, 171, 173, 180–183, 187, 188

## U

Unary, 112, 117, 118, 170

## V

Voltage-averaging, 28, 29  
 Voltage-controlled-oscillator (VCO), 8, 20, 27, 30, 34, 65–70, 72, 74–77, 79, 80, 82–84, 88, 89, 114, 123, 124, 171

## W

Wien-bridge, 16, 23–25