Single-Bit Pseudoparallel Processing
Low-Oversampling Delta–Sigma Modulator
Suitable for SDR Wireless Transmitters

Safar Hatami, Member, IEEE, Mohamed Helaoui, Member, IEEE, Fadhel M. Ghannouchi, Fellow, IEEE, and Massoud Pedram, Fellow, IEEE

Abstract—The oversampling requirement in a delta–sigma modulator (DSM) is considered one of the limiting factors toward its employment in current high-frequency applications, such as wireless software defined radio (SDR) systems. This paper advances that the critical requirement for DSMs is high-frequency processing and not a high-oversampling ratio. A single-bit semiparallel processing structure to accomplish the high-frequency processing is proposed in this paper. Using the suggested low-oversampling digital DSM architecture, high-speed, high-complexity computations, which are normally required for wireless applications, are executed simultaneously. This facilitates the design of embedded SDR multistandard transmitters using commercially available digital processors. The most favorable application of the proposed single-bit DSM is to build an radio frequency transmitter that includes a one-bit quantifier with two-level switching power amplifier for both high linearity and high efficiency. Performance analysis is carried out by using MATLAB simulations, which shows a reduction of the oversampling ratio by a factor of 16 (for a baseline oversampling ratio of 256) with the same signal-to-noise ratio (SNR). The proposed DSM is also implemented on a field-programmable gate array (FPGA) board and its performance is validated by using a code division multiple access signal. The bandwidth of the output signal is increased four times without increasing the processing speed, high-complexity computations, which are normally required for high-speed processing. The oversampling requirement in a DSM discourages its employment in current compute-intensive applications, such as software defined radio systems.

 Emerging applications have encouraged designers to develop highly linear converters with large input bandwidths and switched-mode power supplies. However, achieving this degree of linearity comes at the cost of a large oversampling ratio and, therefore, need for high-speed processing. The oversampling requirement in a DSM discourages its employment in current high-frequency applications, such as software defined radio systems.

 An area-efficient architecture [12] is developed by combining multiple DSMs in parallel, along with analog preprocessing of the input signal and digital postprocessing of the output signals. Through interconnected modulators working in parallel with each running at the same clock, a new parallel processing DSM (PDSM) was proposed in [13]. A time interleaved sigma–delta architecture was used in [14] to increase bandwidth of the converter with a lower hardware complexity.

 In this paper, an alternative approach, also based on parallel processing, is described. However, multiple DSMs are not used. The proposed PDSM implements combined and simplified processing steps for n sequential clocks of a regular DSM (n closed-loop computations). A PDSM that combines n closed loops generates n bits/clock cycle. The highest sampling frequency of the proposed PDSM is now shifted to one multiplexer, which is the same as the sampling frequency of the traditional single-bit DSM. The other processing element of PDSM works n times slower compared with traditional single-bit DSM.

 The favorable application of the proposed PDSM is an RF transmitter that integrates a one-bit quantifier and a two-level switching power amplifier (PA) to attain high linearity. Through the proposed low-oversampling DSM, envelope signals in wireless applications, e.g., orthogonal frequency-division multiplexing (OFDM), are compatible with features such as data converters [3], frequency synthesizers [4], and switched-mode power supplies. However, achieving this degree of linearity comes at the cost of a large oversampling ratio and, therefore, need for high-speed processing. The oversampling requirement in a DSM discourages its employment in current high-frequency applications, such as software defined radio systems.
division multiplexing and code division multiple access (CDMA), can be modulated to two-level signals. These signals can then be amplified with a switch-mode PA. Theoretically, the switch-mode PAs are able to obtain 100% power efficiency. In addition, two-level signals can ideally be processed without any errors (100% linearity.) Therefore, by combining the two-level DSM and switch-mode PA, it is expected that power efficiency and linearity can be simultaneously achieved.

Performance of the proposed technique is validated through MATLAB simulations as well as field-programmable gate array (FPGA) implementation using a CDMA signal.

Section II describes a brief review of a regular oversampling DSM. Section III describes the proposed low-oversampling PDSM. Section IV describes the simulation results and discusses the advantages of the PDSM. The implementation and experimental results are described in Section V. The conclusion of this paper is summarized in Section VI.

II. REVIEW OF OVERSAMPLING LOW-PASS (LP) DSM

This section reviews LP digital DSM theory and provides an example of a third-order digital DSM.

The general structure of a DSM is shown in Fig. 1. The input to the integrator is the difference between the input signal, \( x(t) \), and the quantized output value, \( y(t) \). The quantization noise is represented by the additive term, \( E(t) \). This error is summed in the integrator and then quantized by a two-level quantizer. The output signal, \( y(t) \), is held by DAC for a clock period of \( T_s = 1/f_s \), which yields \( \tilde{y}(t) \). The inherent transfer function of the DAC is \( h(t) \) and relates \( y(t) \), and \( \tilde{y}(t) \) as follows:

\[
\tilde{y}(t) = h(t) \otimes y(t)
\]

where \( \otimes \) is the convolution operator.

The output of a DSM is described in the z-domain by

\[
Y(z) = STF(z)X(z) + NTF(z)E(z)
\]

where \( X(z) \), \( Y(z) \), and \( E(z) \) are the z-transforms of \( x(t) \), \( y(t) \), and \( E(t) \), respectively. The signal transfer function, \( STF(z) \), is applied to the signal at the desired frequency band, whereas the noise transform function, \( NTF(z) \), is applied to the quantization noise to suppress it from the desired band.

A z-domain representation of a third-order LP DSM is shown in Fig. 2. Details about the calculation of the modulator coefficients as well as the seventh-order LP DSM architecture can be found in [15].

For this DSM, the signal and noise transfer functions are given in (3) and (4):

\[
STF(z) = 1
\]

\[
NTF(z) = \frac{(z - 1)(z^2 - 2z + 1)}{(z - 0.6694)(z^2 - 1.5313z + 1.6639)}
\]

The frequency equivalent of (2) is given by (5):

\[
Y(f) = X(f) + NTF(f)E(f)
\]

The frequency domain depiction of this equation is shown in Fig. 3. Fig. 3(a) shows \( |Y(f)| \) and \( |H(f)| \) and (b) shows \( |\tilde{Y}(f)| = |H(f)Y(f)| \). As shown in Fig. 3(a), the shaped noise, \( NTF(f)E(f) \), and signal, \( X(f) \), are repeated at the harmonics of \( f_s \). It is evident from Fig. 3(b) that, among all these signal replicas, the only undistorted signal is at zero frequency. All other replicas are distorted.

III. LOW-OVERSAMPLING PDSM

This section explains the idea behind the proposed low-oversampling architecture for generating two-level delta–sigma output.

For the regular DSM, the sampling frequency of the input signal and clock frequency of the DSM are typically equal (this value is \( f_c \) for previous section). Suppose the sampling frequency of the input signal is \( f_s \) while the clock frequency of the DSM is \( f'_s \), which may not be equal. In addition, assume \( f'_s > f_s \) and, for simplicity of analysis, \( f'_s/f_s \) is a positive integer value, \( N \); therefore, after the elapse of \( N \) clock cycles, the DSM processes one constant digital input. In next sections, \( f'_s \) is the PDSM output rate that is equivalent to the PDSM throughput and output multiplexer.
selective frequency. Also, $f'_s$ can be considered as the effective frequency of PDSM (considering parallel processing). The effective frequency of the PDSM is the alternative name for the sampling frequency for the traditional DSM. The frequency of all processing elements in PDSM is $f_s$ except for the frequency of multiplexer that is $f'_s$.

Table I compares the signals and transfer functions for regular DSM and PDSM. The frequency for the first row is the same for DSM and PDSM. However, they have different frequencies for second, third, and fourth rows of this table.

The DSM associated with Fig. 4 processes a constant input in $N$ clock cycles (in this DSM, $N$ is 2). It is evident from Fig. 4 that, as long as $f_s$ is sufficiently larger than the Nyquist rate of the input signal, $x(t)$, the signal at the baseband is of high quality.

The oversampling ratio formula is given by (6). In (6), $BW$ is the double-sided bandwidth of the signal, and OSR is the oversampling ratio

$$\text{OSR} = \frac{f_s}{BW}. \quad (6)$$

Typically, $f_s$ is eight times greater than the Nyquist rate ($\omega_0$). In contrast, a regular DSM often has an oversampling ratio $\sim 256$ to generate a good quality signal at the output of the DSM. Hence, the sampling frequency of the input signal of the DSM is lower by a factor of 32 or more.

It is clear that an oversampled input signal is not required for the DSM to produce a high-quality output at baseband. However, it is crucial for the DSM to operate at a high frequency, say 256 times the Nyquist rate of the input signal, to stretch the quantization noise in a wide frequency range, and thereby, lower its level in the in-band of the useful signal.

The proposed PDSM takes advantage that the DSM can process constant input samples for $N$ clock cycles. Therefore, a novel architecture that processes $N$ constant samples in parallel by combining $N$ closed-loop processing of a regular DSM is presented. The order of the PDSM is the same as the order of the regular DSM that is used in the PDSM. Herein, $N$ will be referred to as the unrolling factor of the PDSM.

In next sections, a third-order PDSM with $N = 4$ is described; and, finally, the general derivation for the PDSM is provided.

### A. Third-Order and Four-Unrolled PDSM Implementable on FPGA or Application-Specified Integrated Circuit Designs

This section proposes the parallel version of a third-order DSM when the unrolling factor, $N$, is 4. Fig. 5 shows a typical third-order and four-unrolled PDSM architecture. The different components of this architecture are shown in Figs. 7 and 9–12. The input of DSM is sampled with the frequency of $f_s$. All processing elements are working at clock frequency of $f_s$. The frequency of multiplexer, which gives the throughput of PDSM, is $f'_s = 4f_s$. The effective frequency of PDSM is $f'_s = 4f_s$ because of parallel processing.

Fig. 6 shows the parametric version of a third-order DSM, which is also shown in Fig. 2 ($\omega = 2.2e^{-0.05}$, $p = 0.04$, $q = 0.29$, and $r = 0.8$).

It is assumed the signals $a_2[n]$, $a_5[n]$, $a_7[n]$, $a_9[n]$, $x[n]$, and $y[n]$ are the signals at nodes $a_2$, $a_5$, $a_7$, $a_9$, $x$, and $y$, respectively, at time sample $n$. The signals $x[n]$ and $y[n]$ are the input and output signals of the DSM. As it is assumed $N = 4$, the input signal is constant for four consequent clock cycles of $f'_s$, i.e., $x[n] = x[n + 1] = x[n + 2] = x[n + 3]$, where $n$ is a multiple of four.

Equation (7) calculates the values of signals $a_2$, $a_5$, and $a_7$ at time $n + 1$, by using the signal values in the previous time sample, $n$. The signal value $a_9[n]$ is calculated directly from $a_2$, $a_5$, $a_7$, and $x$ at time $n$. The two-level quantizer, $Q$, quantizes $a_9[n]$ into $-1$ or $+1$ at time $n$. The quantized value

### Table I

<table>
<thead>
<tr>
<th>Signal/Transfer Function</th>
<th>DSM</th>
<th>PDSM</th>
</tr>
</thead>
<tbody>
<tr>
<td>$</td>
<td>X(f)</td>
<td>$</td>
</tr>
<tr>
<td>$</td>
<td>E(f)</td>
<td>$</td>
</tr>
<tr>
<td>$</td>
<td>H(f)</td>
<td>$</td>
</tr>
<tr>
<td>$</td>
<td>\text{NTF}(f)</td>
<td>$</td>
</tr>
<tr>
<td>$</td>
<td>Y(f)</td>
<td>$</td>
</tr>
</tbody>
</table>

*Note: $k$ is all nonzero integer values.*

![Fig. 4](image-url)
for the next clock cycle of $f$, is the representation of a third-order LP DSM.

Fig. 5. Typical digital implementation of a third-order and four-unrolled PDSM.

Fig. 6. Representation of a third-order LP DSM.

\[
a_2[n + 1] = a_2[n] + x[n] - y[n]
\]

\[
a_3[n + 1] = a_2[n] + a_3[n] + q a_1[n]
\]

\[
a_1[n + 1] = a_3[n] + p a_1[n]
\]

\[
a_0[n] = r a_2[n] + q a_3[n] + p a_1[n] + x[n]
\]

\[
y[n] = Q(a_0[n]) \quad \text{← Cmp}_0.
\] (7)

Two last equations in (7) correspond to two parts of PDSM in Fig. 5: processing elements (PE0) and the one-bit quantizer (comparator1).

Equation (8) gives the signals $a_2$, $a_3$, and $a_1$ at time $n+2$, by utilizing the signal values at the previous time (previous clock cycle), assuming the input signal, $x$, is constant at time $n$ and $n+1$, i.e., $x[n] = x[n+1]$. To update the signal values for the next clock cycle of $f'$, the basic expressions in (7) are used, and only the time indexes are increased, as shown in

\[
a_2[n + 2] = a_2[n + 1] + x[n + 1] - y[n + 1]
\]

\[
= a_2[n] + 2x[n] - y[n] - y[n + 1]
\]

\[
a_3[n + 2] = a_3[n + 1] + q a_1[n + 1] + a_3[n + 1]
\]

\[
= 2a_2[n] + (p + 1)a_3[n] + 2va_1[n] + x[n] - y[n]
\]

\[
a_1[n + 2] = a_1[n + 1] + a_5[n + 1]
\]

\[
= a_2[n] + 2a_3[n] + (v + 1)a_1[n].
\] (8)

The output signal, $y$, at time $n + 1$ is simply updated as is given in

\[
a_0[n + 1] = par[n + 1] + qa_3[n + 1] + pa_1[n + 1] + x[n + 1]
\]

\[
= (q + r)a_2[n] + (p + q)a_3[n] + (p + v + q)a_1[n] + (r + 1)x[n]
\]

\[
+ (-ry[n])
\]

Second part

\[
= r_1a_2[n] + q_1a_3[n] + p_1a_1[n] + s_11x[n] + s_12y[n] \quad \text{← PE}_1
\]

\[
y[n + 1] = Q(a_0[n + 1]) \quad \text{← Cmp}_1
\] (9)

where $r_1 = q + r$, $q_1 = p + q$, $p_1 = p + v + q$, $s_11 = r + 1$ and $s_12 = -r$.

Two equations in (9) correspond to two parts of PDSM in Fig. 5: processing elements (PE1) and the one-bit quantizer (comparator1), which calculate $y$ at time $n+1$. It is clear from (9) that the process of calculating $a_0[n+1]$ can be divided in two parts as follows: 1) dependent on the signal values of $a_2$, $a_3$, $a_1$, and $x$ at time $n$ and can be processed at time $n$; and 2) dependent on $y[n]$, which is processed by PE1, and its process is started at time $n$. The second part is a two-level value, and its two possibilities can be precalculated and stored in two registers. Once $y[n]$ is ready, the second part is multiplexed from the two precalculated values available in the registers. The first part is computation intensive whose calculation is started at time $n$.

The only computation that depends on $y[n]$ is summation of the precalculated term $s_{12}y[n]$ and the calculation of the second part of (9). Fig. 7 shows the concept of pseudoparallel processing for computing $y[n]$ and $y[n+1]$ together. The total delay associated with the parallel calculation is $T_p = t_m + 3t_s + 2t_c$, where $t_m$, $t_s$, and $t_c$ are parameterized delays of the multiplier, adder, and comparator, respectively. The total calculation delay for regular DSM, $T_r$, is $T_r = 2t_m + 5t_s + 2t_c$. If $t_m = 4t_s$, $t_c = 0.2t_s$ then $T_r/T_p = 13.4/7.4 \approx 2$, which nearly provides a factor of two performance improvement for the parallel processing method for calculating $y[n]$ and $y[n+1]$.

In the next step, all signal values given in (8) and (9) are used to update the signal values for cycle time $n+3$. The updated signal values are given in

\[
a_2[n + 3] = a_2[n] + 3x[n] - y[n] - y[n + 1] - y[n + 2]
\]

\[
a_3[n + 3] = (v + 3)a_2[n] + (3v + 1)a_3[n]
\]

Fig. 7. Typical sequencing diagram for pseudoparallel processing.
\( + (v^2 + 3v)a_7[n] + 3x[n] - 2y[n] - y[n + 1] \)
\( a_7[n + 3] = 3a_2[n] + (3 + v)a_5[n] + (3v + 1)a_7[n] \)
\( + x[n] - y[n] \)
\( a_0[n + 2] = r_2a_2[n] + q_2a_5[n] + p_2a_7[n] + s_{21}x[n] + s_{22}y[n] \)
\( + s_2y[n + 1] \leftarrow P_{E_2} \)
\( y[n + 2] = Q(a_0[n + 2]) \leftarrow Cmp_{P_2} \)
\( (10) \)

where \( r_2 = p + 2q + r, q_2 = 2p + (v + 1)q, p_2 = (v + 1)p + 2vq \)
and \( s_{21} = q + 2r + 1, s_{22} = -q - r, \) and \( s_3 = -r. \)

It is evident from the two last expressions that the output signal at time \( n + 2 \) is obtained from signals at time \( n \) and the output signals \( y[n] \) and \( y[n + 1] \). Therefore, in a digital hardware implementation, the processing of signal \( a_0[n + 2] \) can be started at time \( n \), instead of time \( n + 2 \).

In addition, all significant computations for the \( a_0[n + 2] \) calculation are dependent on the signal values at time \( n \). The only values from times \( n + 1 \) and \( n + 2 \) that contribute in the computing of \( a_0[n + 2] \) are the two-level values, \( y[n] \) and \( y[n + 1] \), of which the two possibilities of the associated products, \( s_{22}y[n] \) and \( s_{33}y[n + 1] \), can be precalculated and stored in two registers. Therefore, once \( y[n] \) and \( y[n + 1] \) are ready, they can be used to evaluate \( a_0[n + 2] \). Two last equations in (10) correspond to two parts of PDSM in Fig. 5: processing elements (PE2) and the one-bit quantizer (comparator), which processes \( y[n + 2] \).

Once again, the basic expressions of (7) are used to compute the output signal at time \( n + 3 \), as given in
\( a_0[n + 3] = r_3a_2[n] + q_3a_5[n] + p_3a_7[n] \leftarrow P_{E_3} \)
\( + s_{31}x[n] + s_{32}y[n] + s_{33}y[n + 1] + s_{34}y[n + 2] \)
\( y[n + 3] = Q(a_0[n + 3]) \leftarrow Cmp_3 \)
\( (11) \)

where \( s_{31} = p + 3q + 3r + 1, s_{32} = -p - 2q - r, s_{33} = -q - r, s_{34} = -r, p_3 = p(3v + 1) + q(v^2 + 3v), r_3 = 3p + q(v + 3), \) and \( q_3 = p(3v + q) + (3v + 1). \)

In the signal derivation in (11), it is supposed that the input signal at time \( n + 3 \) is equal to the value of the signal at time \( n \) \( x[n] = x[n + 3] \). The signal values \( a_0 \) and \( y \) at time \( n + 3 \) are given in the two last expressions of (11). The evaluation process is started at time \( n \) and finished when the two-level values, \( s_{32}y[n], s_{33}y[n + 1], \) and \( s_{34}y[n + 2] \), are available. Two equations in (11) correspond to two parts of PDSM in Fig. 5: processing elements (PE3) and the one-bit quantizer (comparator), which compute \( y[n + 3] \).

In conclusion, the process of calculating four sequential outputs of PDSM can be started simultaneously and accomplished in one clock cycle \( f_p^\prime \). It is evident from (7) to (11) that the path delays of the four sequential outputs are in the same order as the regular DSM, as given in (7).

Figs. 6, 7, 9, and 10 show how to calculate the four sequential outputs of PDSM. However, signals \( a_2, a_5, \) and \( a_7 \) are computed through (12), which is to be used for the next four cycles.
\( - y[n + 3] \)
\( a_5[n + 4] = (4v + 4)a_2[n] + (v^2 + 6v + 1)a_5[n] \)
\( + (4v^2 + 4v)a_7[n] + (6 + v)x[n] \)
\( + (-3 - v)y[n] - 2y[n + 1] - y[n + 2] \)
\( a_7[n + 4] = (v + 6)a_2[n] + (4v + 4)a_5[n] \)
\( + (v^2 + 6v + 1)a_7[n] + 4x[n] - 3y[n] \)
\( - y[n + 1] \).
\( (12) \)

The equations are simply driven by updating the basic expression of (7) for time \( n + 4 \) and utilizing signal values from (8) to (11). A typical implementation of (12) is shown in Fig. 8. This hardware is referred to as the last processor element (last PE) that is part of Fig. 5.

Fig. 5 shows block diagram of a third-order and four-unrolled PDSM architecture contains \( PE_0, PE_1, PE_2, \) and \( PE_3 \) and last PE. The result of an FPGA implementation of the PDSM shown in Fig. 5 is given in next section.

**B. nth-Order and N-unrolled PDSM**

This section proposes the general formulation for the PDSM. Suppose that the digital input sequence is \( x \), where \( x(i) \) is the \( i \)th element of this sequence. The variable \( y \) is the two-level output of DSM, and \( y_a \) is its output before quantization to two levels, \(-1 \) and \( 1 \). The array \( m_{n \times 1} \) is the values of registers in the DSM, where \( n \) is the order of the DSM. The matrices \( A, B, \) and \( C \) describe the coefficients of the DSM.

The expressions in (13) present an \( n \)th-order DSM, when the \( i \)th input is fed to the modulator [15]. A gives the feedback values, whereas \( B \) describes the coefficients from the input and output to the registers. The output value is calculated from the input and the register values by using the coefficient matrix \( C \).

\[ m(i + 1)_{n \times 1} = [A]_{n \times n} \times m(i)_{n \times 1} + [B]_{n \times 2} \begin{bmatrix} x(i) \\ y(i) \end{bmatrix} \begin{bmatrix} 1 \\ 2 \end{bmatrix} \]
\[ y_a(i) = [C]_{1 \times n} [m(i)]_{n \times 1} + x(i) \]
\( (13) \)

where
\[ y = Q(y_a) = \begin{cases} 1 & y_a > 0 \\ -1 & y_a < 0 \end{cases} \]

Let us assume \( f_p^\prime/f_s = N \), meaning that the input of the modulator is constant for each \( N \) clock cycle of \( f_p^\prime \). We want to calculate the feedback values for clock cycle \( N + 1 \) and all output values from the first clock cycle to the \( N \)th clock cycle of \( f_p^\prime \). In the PDSM structure, the output calculations for all \( N \)
sequential outputs are started simultaneously and carried out in one clock cycle of \( f'_s \).

1) First Clock Cycle of \( f'_s \): The following expressions describe signals for the first clock cycle of \( f'_s \):

\[
[m(1)] = [A][m(0)] + [B][x(0)]y(0) \\
y_a(0) = [C][m(0)] + x(0). \tag{14}
\]

2) Second Clock Cycle of \( f'_s \): Assuming \( x(0) = x(1) = 0 \), (13) and (14) are used to calculate signals for the second clock cycle of \( f'_s \), as given in

\[
x(1) = x(0) \\
[m(2)] = [A][m(1)] + [B][x(1)]y(1) \\
= [A][A][m(0)] + [B][x(0)]y(0) + [B][x(1)]y(1) \\
= [A][m(0)] + [A][B][x(0)]y(0) + [B][x(1)]y(1) \\
y_a(1) = [C][m(0)] + [B][x(0)]y(0) + x(1) \\
= [C][A][m(0)] + [C][B][x(0)]y(0) + x(0). \tag{15}
\]

3) Third Clock Cycle of \( f'_s \): For the third clock cycle of \( f'_s \), the signal values can be calculated by using (13)–(15), as given in

\[
x(2) = x(0) \\
[m(3)] = [A][m(2)] + [B][x(2)]y(2) \\
= [A][A][m(0)] + [A][B][x(0)]y(0) + [B][x(0)]y(1) \\
+ [B][x(2)]y(2) \\
= [A][m(0)] + [A][B][x(0)]y(0) + [B][x(0)]y(1) \\
+ [B][x(2)]y(2) \\
y_a(2) = [C][m(2)] + x(2) \\
= [C][A][m(0)] + [A][B][x(0)]y(0) + [B][x(0)]y(1) \\
+ x(0) \\
= [C][A][m(0)] + [A][B][x(0)]y(0) + x(0) \\
+ [C][B][x(0)]y(1) + x(0). \tag{16}
\]

Therefore, \( y_a(N) \) and \( m(N) \), the output and feedback values at the \( N \)th clock period of \( f'_s \), respectively, can be expressed as

\[
m(N) = [A][m(0)] + [A][B][x(0)]y(0) \\
+[A][B][x(0)]y(1) + \ldots + [A][B][x(0)]y(N-2) \\
+[B][x(0)]y(N-1) + [B][x(0)]y(N) \\
[y_a(N)] = [C][A][m(0)] + [C][A][B][x(0)]y(0) \\
+[C][B][x(0)]y(1) + [C][B][x(0)]y(2) + \ldots + [C][B][x(0)]y(N-2). \tag{17}
\]

The signals that should be calculated in the PDSM for \( N \) clock cycles of \( f'_s \) are the output signal values, \( y_a(i) \) \( (i = 1, \ldots, N) \), and the feedback values at the \( N \)th clock cycle of \( f'_s \), \( m(N) \), which is used for cycle \( N + 1 \). Considering the two last equations, it is evident that \( y_a(i) \) and \( m(N) \) can be rewritten as given in (18), where \( w_i, b_i, \epsilon_i \), and \( d_i \) are calculated from.

\[
[m(N)] = \omega_1m_1(0) + \omega_2m_2(0) + \ldots + \omega_nm_n(0) + ax(0) \\
+b_1y(0) + b_2y(1) + \ldots + b_Ny(N-1) \\
+\left( d_1y(0) + d_2y(1) + \ldots + d_{N-1}y(i-1) \right) \tag{18}
\]

The calculations of \( y_a \) and the last feedback values \( m(N) \) are divided each into two expressions. The first expression only depends on the feedback values, \( m(0) \), which can be processed in one clock cycle of \( f'_s \). Hence, to evaluate \( N \) sequential outputs of the DSM \( (N \) bits), for each bit, there are \( n \) multiplications for the feedback coefficients and one multiplication for the inputs that can be calculated in parallel. The results of the \( n+1 \) multiplications must be added by using \( n \) adders. The same situation is valid for the computation of the last feedback values.

In the second expression, \( y(i) \) is a two-level value, so its multiplication is simple. The results for the two expressions must finally be added together. The calculation of \( y_a(i) \) requires \( y(i-1) \) from the last \( y_a \) calculation. Fig. 9 shows a hardware implementation of a PDSM, based on (14)–(18). The architecture is an extension of the third-order and four-unrolled PDSM shown in Fig. 5. It shows that \( N \) processor elements calculate \( N \) outputs in parallel. One processor calculates the states of the registers for the cycle \( N + 1 \). The frequency of input sampling and for processing elements is \( f_s \). The PDSM output rate, which is equivalent to the PDSM throughput and output multiplexer selection frequency, is \( f'_s \). \( f'_s \) can be called the effective frequency of PDSM, which considers parallel processing.

IV. SIMULATION RESULTS

This section gives simulation results for the proposed PDSM and compares these results with the conventional DSM. A hardware implementation of the DSM is presented in this section. The suggested architecture is implementable with current digital complementary metal-oxide-semiconductor technology and can be utilized in RF wireless applications.
A. Simulation Results for LP DSM

The criterion for comparison is the signal-to-noise ratio (SNR) shown in (19), where $MS$ is the mean square. SNR is defined as the ratio of the in-band signal power to the in-band and out-of-band noise power of modulated signal

$$\text{SNR} = 10 \log \left( \frac{MS(\text{Signal})}{MS(\text{Noise})} \right). \quad (19)$$

A PDSM and regular DSM are implemented in MATLAB for first- to seventh-order DSMs. Simulations using a CDMA modulated signal are carried out and the results are shown in Table II for three different DSM orders for both the regular DSM and the proposed PDSM. $N$ is the unrolling factor of the PDSM. The frequency column is the DSM and PDSM processing element clock frequency of processing, $f_s$. The multiplexer selection frequency is $f'_s = 5.12$ MHz and the frequency bandwidth is 2.048 kHz. The frequency of processing for PDSM changes from 0.131 to 2.097 MHz but its throughput is 2.097 MHz. For SNR calculation given in (19), the single-sided bandwidth for in-band signal and out-of-band noise are 20 kHz. For example, this table shows that the SNR of modulated signal for the second-order DSM is 64.7 dB and reduces $\sim$15 dB for each folding of OSR (experiments 11–15). It also shows that SNR $\sim$65 dB for second-order PDSM with $N \times \text{OSR} = 512$ for $N = 2, 4, 8,$ and $16$ (experiments 16–20). Similar results are shown in Table II for the second-, third-, and fifth-order DSM and PDSM. Fig. 10(a) and (b) shows the spectrum of the modulated CDMA signals at the output of the DSM for experiments 11 and 20, respectively. While PDSM allowed reducing the sampling frequency by 16 times by using parallel processing, the SNR remained at about the same level $-64.7$ dB for DSM against 67.6 dB for PDSM.

V. EXPERIMENTAL VALIDATION USING DS TRANSMITTER

A gigahertz PDSM-based transmitter is developed, prototyped, and used to validate the approach proposed in this paper. Fig. 11 shows the block diagram of the demonstrator. The PDSM transmitter is implemented in two blocks. The baseband signal processing part is implemented using an FPGA block. The modulation and up-conversion is implemented using a high-speed dedicated logic stage.

The two third-order DSMs and PDSMs shown in Figs. 5 and 6 are implemented on a Stratix II EP2S60 DSP development board [16] and tested with a CDMA signal. The baseband in-phase (I) and quadrature (Q) signals are read from two on-board memories and fed through the LP DSMs/PDSMs. Three multiplexers are used for up-conversion and in I/Q modulation at carrier frequency. The binary RF output signal is fed to a vector signal analyzer, which is used to capture, filter, and analyze the signal.

The main advantage of PDSM is to achieve higher SNR output signal using lower processing frequency compared with a regular DSM. One of the most favorable applications of the proposed single-bit PDSM is to make an RF transmitter that includes a one-bit quantizer delta–sigma and two-level switching PA, which results in a high-efficiency and high-linear transmitter. A two-level switching PA Class $D, E, F, F^{-1}$ or $S$ can be driven with the two-level output of DSM [1], [2], [5], [6]. Fig. 12 shows a block diagram of the setup used to evaluate the performance of the PDSM-based transmitter. Fig. 13 shows a photo of the measurement setup for a prototype DSM/PDSM-based RF transmitter.

The unrolling factor of the implemented PDSM is selected to be four. The PDSM and DSM are fed by CDMA signals.
processing, the PDSM allows for an increase of the modulation bandwidth by a factor of four compared with DSM, while maintaining a comparable noise shaping performance. The SNRs of the output signals for both cases are approximately the same level, 49 dB for DSM and 47 dB for PDSM, and are shown in Table III.

### A. Area and Power

Table IV shows the evaluation of the resources occupied in the FPGA, in terms of number of logic cells for gates, register, and arithmetic logic units (ALUs). The improvement in performances in the PDSM architecture ($N = 4$) comes with an increase in the resources required for implementation. It is shown that the resources are increased about three times for $N = 4$. In general, based on the methodology in Section III-A, the architecture of a PDSM with unrolling factor $N$ is obtained by unrolling structures of $N$ regular DSM. As shown in (7)–(12) the summation and multiplication operations are simplified and optimized. Therefore, hardware of a PDSM with unrolling factor $N$ is smaller than $N \times A_{\Delta \Sigma}$, where $A_{\Delta \Sigma}$ is the area of regular DSM with same order of noise shaping. The FPGA area for $N = 4$ and third-order PDSM as shown in Table IV is $\sim(3/4) \times N \times A_{\Delta \Sigma}$.

The power consumption of the development board and multiplexer is in the order of 100 mW. This includes PDSM components and other unused components on FPGA development board. The power consumption of PA in a PDSM-based transmitter is in the order of 10 W. Therefore, power consumption of PDSM is negligible compared with total power consumption of the transmitter.

### B. Comparison

Table V compares different single-bit PDSM structures. For comparison, it is assumed the order of different delta–sigma architectures is the same. References [12] and [13] implemented PDSM using analog circuits in [10] and [11] only reported simulation results. As each design is implemented on different technology, design areas are compared parametrically. The power consumption values are not available for every referenced design. The first row of Table V considers a regular DSM with processing frequency $f'_s$ and area $A_{\Delta \Sigma}$. The throughput of different delta–sigma architectures is

---

**Figure 10.** Spectrum of third-order DSMs for a CDMA input signal: (a) DSM: $f'_s = 2.097$ MHz, OSR = 512, and SNR = 64.7 dB; (b) PDSM: $f'_s = 2.097$ MHz and $f_s = 0.131$ MHz, OSR = 32, SNR = 67.6 dB, and unrolling factor of 16.

**Figure 11.** Block diagram delta–sigma-based transmitter.

**Figure 12.** Block diagram of setup to test PDSM-based transmitter.

---
Fig. 14. Spectrum of the output signal (the signal BW for the PDSM is four times the signal BW for the DSM). (a) Third-order PDSM with an unrolling factor of 4. (b) Third-order DSM.

TABLE III

SNR COMPARISON OF THE THIRD-ORDER DSM AND PDSM (THE SIGNAL BW FOR PDSM IS FOUR TIMES THE SIGNAL BW FOR DSM)

<table>
<thead>
<tr>
<th>Structure</th>
<th>Processing Clock (MHz)</th>
<th>SNR (dB)</th>
<th>BW (kHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DSM</td>
<td>25</td>
<td>49</td>
<td>400</td>
</tr>
<tr>
<td>PDSM</td>
<td>25</td>
<td>47</td>
<td>1600</td>
</tr>
</tbody>
</table>

It is worthwhile to mention that there are multibit DSM structures that lower required processing frequency. However, the focus of this paper is single-bit DSM that is most applicable in RF transmitter with one-bit quantizer and two-level switching mode PA. For example, multibit quantizer delta–sigma is a structure that insures linearity with a lower processing frequency and a lower OSR value compared with regular DSM. The multistage noise shaping structure is also an alternative delta–sigma structure that is simple for implementation and it is unconditionally stable [15].

VI. CONCLUSION

A new DSM architecture was introduced. This structure performed delta–sigma modulation with a smaller oversampling rate. The proposed architecture used the concept of parallel processing to achieve the effect of oversampling without the need for a high-sampling frequency. The analysis presented was general and was applicable for LP and bandpass DSMs. The proposed structure was validated through MATLAB simulation. Simulation results showed that for a DSM with OSR = 256, the proposed structure was able to fold the required OSR 16 times while maintaining the same SNR. A 1-GHz carrier frequency transmitter with a CDMA signal was implemented on FPGA using pseudoparallel processing low-oversampling DSM and regular DSM. The proposed architecture was able to increase the bandwidth of the output signal four times without increasing the processing frequency while producing the same quality of output signal.
REFERENCES


Safar Hatami (M’11) received the B.Sc. degree in electronics engineering from the Sharif University of Technology, Tehran, Iran, in 2000, and the M.Sc. degree in electrical engineering from the University of Tehran, Tehran, in 2003, the M.Sc. degree in electrical engineering (communications) from the University of Calgary, Calgary, AB, Canada, and the Ph.D. degree in electrical engineering and the M.Sc. degree in computer engineering from the University of Southern California, Los Angeles, CA, USA, in 2010 and 2011, respectively.

He was with IBM Hudson Valley, Research Park, NY, USA, in 2008. He was with the IBM T.J. Watson Research Laboratory, Yorktown, NY, USA, in 2010. He is currently a Senior Staff Scientist of design with Broadcom Corporation, Irvine, CA, USA. He has filed four patents and disclosures and published over 50 papers in highly valued journals and conference proceedings. His current research interests include signal processing, high speed transmitter design for SDR applications, timing analysis, and performance variation modeling of VLSI circuits.

Mohamed Helaoui (’06–M’09) received the M.Sc. degree in communications and information technology from École Supérieure des Communications de Tunis, Tunis, Tunisia, in 2003, and the Ph.D. degree in electrical engineering from the University of Calgary, Calgary, AB, Canada, in 2008.

He is currently an Assistant Professor with the Department of Electrical and Computer Engineering, University of Calgary. His current research interests include digital signal processing, power efficiency enhancement for wireless transmitters, switching mode power amplifiers, six-port receiver and advanced transceiver design for software defined radio, and millimeter-wave applications. His research activities have led to over 90 publications and seven patents (pending).

Dr. Helaoui is a member of the COMMTTAP Chapter in the IEEE Southern Alberta Section.

Fadhel M. Ghannouchi (F’07) is currently a Professor, a AITF/Canada Research Chair, and the Director with the iRadio Laboratory, Department of Electrical and Computer Engineering, University of Calgary, Calgary, AB, Canada. His current research interests include RF and wireless communications, nonlinear modeling of microwave devices and communications systems, design of power- and spectrum-efficient microwave amplification systems, and design of SDR systems for wireless and satellite communications applications. His research has led to over 60 refereed publications, eight U.S. patents and seven patents applications, and four books.

He is the IEEE-MTT-S Emeritus Distinguished Microwave Lecturer.

Massoud Pedram (F’01) received the Ph.D. degree in electrical engineering and computer sciences from the University of California, Berkeley, CA, USA, in 1991.

He is the Stephen and Etta Varra Professor of electrical engineering with the University of Southern California, Los Angeles, CA, USA. He holds 10 U.S. patents and has published 4 books, 12 book chapters, and more than 130 archival and 300 conference papers. His research ranges from low power electronics, energy-efficient processing, and cloud computing to photovoltaic cell power generation, energy storage, and power conversion, and from RT-level optimization of VLSI circuits to synthesis and physical design of quantum circuits.

Dr. Pedram was a recipient of six conference and two IEEE Transactions Best Paper Awards. He was a recipient of the 1996 Presidential Early Career Award for Scientists and Engineers, a fellow of the ACM Distinguished Scientist, and he is currently the Editor-in-Chiefs of the ACM Transactions on Design Automation of Electronic Systems and the IEEE Journal on Emerging and Selected Topics in Circuits and Systems. He has served on the Technical Program Committee of a number of premiere conferences and was the Founding Technical Program Co-chair of the 1996 International Symposium on Low Power Electronics and Design and the Technical Program Chair of the 2002 International Symposium on Physical Design.