SEGMENTED MISMATCH-SHAPING D/A CONVERSION

Asaf Fishov, Eric Siragusa, Jared Welz, Eric Fogleman, Ian Galton*

University of California, San Diego, 9500 Gilman Drive, La Jolla, CA, 92093-0407, USA
*Silicon Wave, Inc. 6256 Greenwich Drive Suite 300, San Diego, CA 92122

ABSTRACT

Tree-structured mismatch-shaping DACs have recently been used as enabling components in state-of-the-art ΔΣ ADCs to suppress signal-band error arising from mismatches among fabricated DAC components. This paper presents a segmented version of the encoder with significantly reduced hardware complexity. The segmented architecture follows from a result that any digital encoder for which the output values sum to the input value can be implemented using a tree-structured encoder.

1. INTRODUCTION

Many high-performance ΔΣ data converters employ dynamic element-matching (DEM) techniques to spectrally shape *DAC noise*, which arises from static mismatches in the step sizes between adjacent output levels of multibit DACs. Most DEM DACs belong to a class of DACs that consist of a device called a *unit-element digital encoder* followed by a bank of one-bit DACs, where each one-bit DAC has a weighting (i.e., step-size) of one least-significant bit (LSB). The digital encoder maps each input sample, $x[n]$, to the output such that the sum of the encoder outputs equals $x[n]$. The purpose of the digital encoder is to cause the DAC noise to be white or spectrally shaped, as required by the application. Tree-structured DACs are area-efficient members of this class and have been used as enabling components in state-of-the-art ΔΣ and pipelined ADCs. Nevertheless, as the number of DAC bits, b, increases, the circuit area required by the digital encoder, combined with the 2^b one-bit DACs required in the DAC bank, becomes prohibitively large for any DAC in this class.

Recently, a technique that overcomes this limitation has been presented that combines two DACs with different weights to form a *segmented* DEM DAC [1]. This paper extends the technique presented in [1] by showing how the unit-element tree-structured encoder can be used to design segmented tree-structured encoders. Additionally, it is shown that any segmented or unit-element encoder can be implemented as a tree-structured encoder. It follows that the tree-structured encoder provides a general tool for the design of any arbitrary segmented or unit-element encoder. To illustrate the benefits of segmentation, simulation results and implementation details for a 123-dB dynamic range audio ΔΣ DAC employing a segmented tree-structure are provided. The procedure for synthesizing a segmented tree structure from an original tree structure, and a detailed proof verifying the claim that any digital encoder can be implemented as a tree-structured encoder are also provided.

2. BACKGROUND

A 9-level version of the unit-element tree-structured DAC is shown in Figure 1a. It consists of a tree-structured encoder followed by a bank of eight one-bit DACs, each with a weight of Δ, where Δ equals one LSB. The analog outputs of the one-bit DACs are summed to generate the overall output, $y[n]$. The input sequence, $x[n]$, is restricted to the set {-4, ..., 4}. Each one-bit DAC generates an analog output given by $\Delta/2+e_{h_i}$ if its input equals +½, and $-\Delta/2+e_{l_i}$ if its input equals −½, where Δ is the nominal step-size, and e_{l_i} and e_{h_i} are the low and high one-bit DAC errors of the i^{th} DAC. The one-bit DAC errors arise from component mismatches and are assumed to be static. As shown in [2], the overall DAC output has the form $y[n] = \alpha x[n] + \beta + e[n]$, where α is a constant gain, β is a constant offset, and $e[n]$ is *DAC noise* that depends upon the input sequence, the details of the digital encoder, and the one-bit DAC errors.

The encoder consists of *switching blocks*, labeled $S_{k,r}$, with $1 \le k \le 3, 1 \le r \le 4$ in Figure 1a, where k denotes the layer number and r denotes the position of the switching block within the layer. Each switching block performs the operations shown in Figure 1b where $s_{k,r}[n]$ is a sequence generated within $S_{k,r}$. These operations are expressed explicitly by the equations

$$x_{k-1,2r}[n] = \frac{x_{k,r}[n] - s_{k,r}[n]}{2} \tag{1}$$

and

$$x_{k-1,2r-1}[n] = \frac{x_{k,r}[n] + s_{k,r}[n]}{2}. \tag{2}$$

If, for each k and r, $s_{k,r}[n]$ is chosen such that the outputs of switching block $S_{k,r}$ are in the set $\{-2^{k-2}, ..., 2^{k-2}\}$ and sum to $x_{k,r}[n]$, then the digital encoder is said to obey the *number conservation rule*. Provided the number conservation rule is satisfied, the sum of the output values of the tree-structured encoder is guaranteed to equal $x[n]$. As shown in [2], the DAC noise is given by

$$e[n] = \sum_{k=1}^{3} \sum_{r=1}^{2^{3-k}} \Delta_{k,r} s_{k,r}[n], \tag{3}$$

where the $\Delta_{k,r}$ coefficients are constants that depend only on the static one-bit DAC errors. It follows from (3) that a sufficient condition for the DAC noise $e[n]$ to have a specific spectral property (e.g., first-order shaped) is that each $s_{k,r}[n]$ must possess that spectral property and be uncorrelated from the $s_{k,r}[n]$ sequences of the other switching blocks. As shown in [3], simple switching blocks exist that satisfy the number conservation rule and implement first- and second-order shaping.

Given any arbitrary digital encoder, Section 5 shows how to select a tree structure and obtain the switching sequences which can be used to implement that encoder. It follows that the results described above can be applied to any digital encoder.

3. THE SEGMENTED TREE STRUCTURE TOPOLOGY

In all of the known non-segmented mismatch-shaping DACs, including tree-structured DACs, circuit area increases exponentially with the number of bits. Consequently, such DACs are rarely used for more than 5-bit conversion. However, in many

mismatch-shaping DAC applications, using higher than 5-bit conversion offers significant benefits. For example, in audio $\Delta\Sigma$ DACs, increasing the number of bits relaxes the requirements on the analog reconstruction filters, which tend to consume a significant portion of the total power. Segmented digital encoders make it practical to increase the number of bits in these applications because the digital encoder size increases linearly with the number of bits.

Figure 2 shows a 257-level segmented tree-structured DAC. It consists of switching blocks labeled $S_{k,r}$, where k denotes the layer number and r denotes the position of the switching block within the layer. Switching blocks in layers 1 through 4 function in the same manner as those in a unit-element tree structure, and hardware implementations for these switching blocks are detailed in [3]. Switching blocks in layers 5 through 8 requantize their input value to an even integer, which is then sent to the top output. A divide-by-two operation is performed on this requantized output sequence before it is input to the next layer. For these switching blocks, the bottom output equals $-\varepsilon_{k,r}[n]$, where $\varepsilon_{k,r}[n]$ is the requantization error sequence generated by requantizing the input value. Without segmentation, the digital encoder would consist of 255 switching blocks and 256 one-bit DACs. Segmenting reduces these numbers to 23 switching blocks and 24 weighted one-bit DACs. Figure 3 shows a hardware-efficient implementation of a switching block that requantizes its input to evens and then divides the top output by two. It can be used to directly implement layers 5 through 8 of the segmented tree-structured encoder of Figure 2. The switching block employs extra-LSB encoding of its input and output sequences whereby $x_{k,r}[n]$ consists of k+1 bits, denoted by $x_{k,r}^{(i)}[n]$, (i = 0, ..., k) and $x_{k,r}[n] = 2^{k-1}x_{k,r}^{(k)}[n] + 2^{k-2}x_{k,r}^{(k-1)}[n] + \ldots + x_{k,r}^{(1)}[n] + x_{k,r}^{(0)}[n] - 2^{k-1}$. The sequencing logic and XOR gate determine $\varepsilon_{k,r}[n]$ according to (5) and depending on the parity of the input sequence. The sign and magnitude of $\varepsilon_{k,r}[n]$ are represented by $q_{k,r}[n]$ when the input sequence is odd. Therefore, when $\varepsilon_{k,r}[n]$ equals 1, both output bits of $x_{k-1,2r}[n]$ are set high, and when $\varepsilon_{k,r}[n]$ equals –1, both bits are set low. For even inputs, $\varepsilon_{k,r}[n]$ is zero, corresponding to one of these bits set high and the other set low. The sequence $d_{k,r}[n]$ represents a random dither sequence and is used to eliminate spurious tones in the shaped DAC noise.

An application that typically employs a digital encoder to provide DAC noise shaping is a high-performance audio $\Delta\Sigma$ DAC. These DACs generally consist of a digital $\Delta\Sigma$ modulator, a digital encoder followed by a bank of one-bit DACs, and an analog reconstruction filter. As mentioned above, the reconstruction filter tends to consume a significant portion of the total power. Simulation results for an 8-bit first-order digital $\Delta\Sigma$ modulator with an OSR of 512 and the segmented tree-structured DAC of Figure 2 indicate that 123 dB of dynamic range is achievable in a 24 kHz bandwidth. The $\Delta\Sigma$ modulator is dithered to eliminate spurious tones in the shaped quantization noise. Static DAC errors are chosen randomly from a Gaussian distribution with a standard deviation of 1% of the nominal LSB step size, with non-LSB weighted DACs having a standard deviation proportional to the square root of their weight. A power spectral density plot of the unfiltered output with the input signal removed is shown in Figure 4a. The total integrated shaped quantization noise and DAC noise from 24 kHz to 100 kHz is less than –110 dB. This low level of out-of-band noise results in a smooth output waveform without a reconstruction filter, as is shown in Figure 4b.

4. SYNTHESIS OF THE SEGMENTED TREE STRUCTURE

The process for generating a segmented tree structure from a unit-element tree structure is described below using the 9-level tree structure of Figure 1a. Although the resulting 5-level segmented tree structure does not provide a reduction in hardware complexity when compared to a 5-level unit-element tree structure, it does provide a simple vehicle with which to explain the process of generating a segmented tree-structured DAC from a unit-element tree-structured DAC. As mentioned above and detailed below, tree structured encoders can be used to construct any segmented DEM encoder.

Suppose $s_{3,1}[n]$ is given by

$$s_{3,1}[n] = x[n] + 2\varepsilon_{3,1}[n], \tag{4}$$

where $\varepsilon_{3,1}[n]$ is chosen such that

$$\varepsilon_{3,1}[n] = \begin{cases} \mp 1, & x[n]\ odd \\ 0, & x[n]\ even, \end{cases} \tag{5}$$

and the flexibility in (5) when $x[n]$ is odd is exploited such that $\varepsilon_{3,1}[n]$ has the spectral properties desired of the DAC noise. Substituting (4) into (1) and (2) gives

$$x_{2,1}[n] = x[n] + \varepsilon_{3,1}[n] \text{ and } x_{2,2}[n] = -\varepsilon_{3,1}[n]. \tag{6}$$

In order for $S_{3,1}$ to obey the number conservation rule, equations (4) and (5) imply that $x[n] \in \{-2, -1, 0, 1, 2\}$, a restriction that is a result of forcing $x[n]$ to be contained only in the top output $x_{2,1}[n]$. Furthermore, (4) and (5) imply that $x_{2,1}[n]$ will be even for all n. Alternatively, one can think of $S_{3,1}$ as having requantized $x[n]$ to an even integer with $\varepsilon_{3,1}[n]$ being the requantization error sequence. Either way, the input to $S_{2,1}$ will always be even and $s_{2,1}[n]$ can be set to zero for all time without violation of the number conservation rule. It then follows from (1) and (2) that the only function of $S_{2,1}$ is to perform a divide-by-two operation. Moreover, the top and bottom outputs of $S_{2,1}$ will be equal and

$$x_{1,1}[n] = x_{1,2}[n] = \frac{x[n] + \varepsilon_{3,1}[n]}{2}. \tag{7}$$

This implies that $s_{1,1}[n]$ and $s_{1,2}[n]$ can be equal for all n without violation of the number conservation rule. In this case, the top outputs of $S_{1,1}$ and $S_{1,2}$ will also be equal. The same will be true for the bottom outputs. This redundancy is exploited by eliminating the $S_{2,1}$ switching block and replacing it with a divide-by-two operation, indicated by an arrow with a ½ next to it in Figure 5. Since the top and bottom outputs of $S_{1,1}$ and $S_{1,2}$ are equal for all time (since their inputs and $s_{k,r}[n]$ sequences are equal), their functionality can be combined into one switching block whose outputs are connected to one-bit DACs with a step size of 2Δ instead of Δ. This eliminates the need for $S_{1,2}$ and its corresponding one-bit DACs, so they are removed. This choice of $s_{3,1}[n]$ also yields a hardware reduction in the lower branch of the encoder. As shown above, $x_{2,2}[n] = -\varepsilon_{3,1}[n] \in \{-1, 0, 1\}$. Since $x_{2,2}[n]$ is a 3-level sequence, it can be applied directly to a switching block in the final layer of the tree structure without violation of the number conservation rule. Choosing $s_{2,2}[n] = x_{2,2}[n]$ causes $x_{1,3}[n] = x_{2,2}[n]$, while forcing the input to $S_{1,4}$ to be zero for all time. $S_{1,4}$ and its associated one-bit DACs are then eliminated without affecting the overall output. In this case $S_{2,2}$ only serves to connect the bottom output of $S_{3,1}$ to the input of $S_{1,3}$. $S_{2,2}$ is then eliminated and these nodes are connected directly. The end result is a 5-level segmented tree-structured DAC and is shown in Figure 5. Since it is not possible to further segment the tree

structure in Figure 5, it is referred to as a *fully-segmented* tree-structured encoder.

As mentioned above, switching block $S_{3,1}$ in Figure 5 can be thought of as having requantized each value of the input sequence to an even integer. For adequately sized unit-element tree structures, it is possible to requantize to other integers besides 2. An example of a segmented tree structure that uses a switching block to requantize its input sequence to a multiple of 4 instead of 2 is shown in Figure 6. This 33-level segmented tree-structured DAC was derived from a 65-level unit-element tree-structured DAC by restricting the input range to 33 levels from 65 levels and choosing $s_{6,1}[n]$ to be of the form given in (4). The steps outlined above are then applied to realize the encoder in Figure 6. In this case, since the input is requantized to a multiple of 4 and sent out the top output of $S_{6,1}$, the next two switching blocks can have $s_{k,r}[n]$ sequences equal to zero for all time. These switching blocks are represented as a divide-by-four operation in Figure 6. For this example, the requantization error sequence $\varepsilon_{4,1}[n] \in \{-3, -2, ..., 2, 3\}$. Since it is possible to further segment sub-$DAC_{3,1}$ and sub-$DAC_{3,2}$, the overall DAC is referred to as a *partially-segmented* tree structure.

As shown in the next section, tree-structured encoders can be used to synthesize any segmented or unit-element mismatch-shaping DAC.

5. THEORETICAL DETAILS

The purpose of this section is to show that any digital encoder can be implemented by a tree-structured encoder. The first step is to define a *unit-output* digital encoder. An L+1-level unit-output digital encoder, U, maps an input sequence, $x[n] \in \{0, 1, ..., L\}$, into L output bits, $u_i[n] \in \{0, 1\}$, i=1, ..., L, such that

$$x[n] = \sum_{i=1}^{L} u_i[n]. \tag{8}$$

Let U^L be the set of all L+1 level unit-output encoders. The first claim shows that any unit-output digital encoder can be implemented using a tree structure. The second claim extends this result to segmented encoders.

Claim 1: Suppose a tree-structured encoder T and unit-output encoder U are given. The outputs $x_{0,i}[n] = u_i[n]$ for $i = 1, 2, ..., 2^b$ when

$$s_{k,r}[n] = \sum_{i=(r-1)2^k+1}^{(r-1)2^k+2^{k-1}} (u_i[n] - u_{i+2^{k-1}}[n]). \tag{9}$$

Proof: Let encoder $U \in U^{2^b}$ with input sequence $x[n] \in \{0, 1, ..., 2^b\}$ be given. Let $p(k)$ be the following statement:

If $s_{k,r}[n]$ is defined as in(9), with $s_{0,r}[n] \equiv 0$, $x_{k,r}[n]$ is defined as in (1) and (2), and $x_{b,1}[n] = x[n]$, then

$$x_{k,r}[n] = \sum_{i=(r-1)2^k+1}^{r2^k} u_i[n] \tag{10}$$

for each $r \in \{1, 2, ..., 2^{b-k}\}$.

Claim: $p(k)$ holds for each $k \in \{0, 1, ..., b\}$. This will be proven by induction.

Initial Step: Let $k = b$. Then $x_{b,1}[n] = x[n] = \sum_{i=1}^{2^b} u_i[n]$, where the first equality is true by the hypothesis of $p(k)$ and the second equality is true by the definition given above.

Induction step: Next suppose $p(k)$ holds for some $k = k' < b$. Then for $k = k' - 1$, equation (1) gives

$x_{k'-1,2r-1}[n] = \dfrac{x_{k',r}[n] + s_{k',r}[n]}{2}$. By the induction hypothesis, (10) holds for $k = k'$. Substituting (10) for $x_{k',r}[n]$ and (9) for $s_{k',r}[n]$ in the previous equation gives

$x_{k'-1,2r-1}[n] = \sum_{i=(r-1)2^{k'}+2^{k'-1}}^{(r-1)2^{k'}+2^{k'-1}} u_i[n]$, which is equal to (10) with k and r replaced by $k'-1$ and $2r-1$ respectively. By the same analysis, but subtracting $s_{k',r}[n]$, it follows that

$x_{k'-1,2r}[n] = \sum_{i=(r-1)2^{k'}+2^{k'-1}+1}^{r2^{k'}} u_i[n]$, which is equal to (10) with k and r replaced by $k'-1$ and $2r$, respectively. Then by induction, (10) holds for each k and r, proving the claim.

Furthermore,

$$x_{k,r}[n] = \sum_{i=(r-1)2^k+1}^{r2^k} u_i[n] \in \{0, 1, ..., 2^k\}, \tag{11}$$

and

$$x_{k,r}[n] = \sum_{i=(r-1)2^k+1}^{r2^k} u_i[n] = \sum_{i=(r-1)2^k+1}^{(r-1)2^k+2^{k-1}} u_i[n] + \sum_{i=(r-1)2^k+1}^{(r-1)2^k+2^{k-1}} u_{i+2^{k-1}}[n]$$
$$= x_{k-1,2r-1}[n] + x_{k-1,2r}[n]. \tag{12}$$

Together, (11) and (12) imply that $x_{k,r}[n]$ satisfies the number conservation rule for each k and r in T. The resulting $x_{k,r}[n]$ and $s_{k,r}[n]$ sequences define the operation of a tree-structured encoder with $x_{0,i}[n] = u_i[n]$ and i = 1, 2, ..., 2^b. Thus when $s_{k,r}[n]$ is defined as in (9) and U and T receive the same arbitrary input sequence, $x_{0,i}[n] = u_i[n]$ and i = 1, 2, ..., 2^b. It follows that encoder U can be implemented using tree structure T.

Claim 1 will now be extended to include segmented encoders.

Let S be an L-line segmented encoder with associated weight vector $\underline{\alpha} = \{\alpha_1, \alpha_2, ..., \alpha_L\}$, where $\underline{\alpha} > 0$. S maps an associated class of non-negative input sequences, $x[n]$, into L binary sequences, $s_i[n] \in \{0,1\}$, i=1, 2, ..., L. such that $x[n] = \sum_{i=1}^{L} \alpha_i s_i[n]$.

Claim 2: Suppose an L-line segmented encoder, S, is given. Define a_i and b as: $a_i = \sum_{j=1}^{i-1} \alpha_j$, $i = 1, 2, ..., L+1$

and $b = \lceil \log_2 a_{L+1} \rceil$. Then there exists a tree-structured encoder, $T \in T^{2^b}$ such that $x_{0,r}[n] = s_i[n]$ for each $r \in \{a_i + 1, a_i + 2, ..., a_i + \alpha_i\}$, i = 1, 2, ..., L, for any input sequence $x[n]$ in the given class of input sequences.

Proof: Select $T \in T^{2^b}$. By the above claim, the outputs of T can be defined as $x_{0,r}[n] \equiv 0$ for $a_{L+1} < r \le 2^b$ and $x_{0,r}[n] = s_i[n]$ for each $r \in \{a_i + 1, a_i + 2, ..., a_i + \alpha_i\}$ and i = 1, 2, ..., L. By these definitions, T has the desired properties.

Acknowledgements: This work was supported by the National Science Foundation under Award 0073552.

6. REFERENCES

[1] R. Adams, K. Q. Nguyen, "A 113-dB SNR Oversampling DAC with Segmented Noise-Shaped Scrambling", *IEEE Journal of Solid State Circuits*, vol. 33, no. 12, pp. 1871-1878, December 1998.

[2] I. Galton, "Spectral Shaping of Circuit Errors in Digital-to-Analog Converters", *IEEE Transactions on Circuits and Systems II*, vol.44, no. 10, pp.808-817, October 1997.

[3] J. Welz, I. Galton, E. Fogleman, "Simplified Logic for First-Order and Second-Order Mismatch-Shaping Digital-to-Analog Converters", *IEEE Transactions on Circuits and Systems II*, accepted for publication.

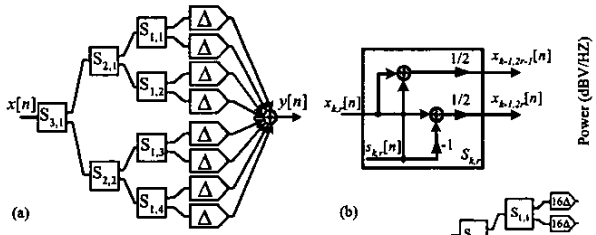

Figure 1. a) 9-level unit-element tree-structured DAC. b) The general form of all switching blocks

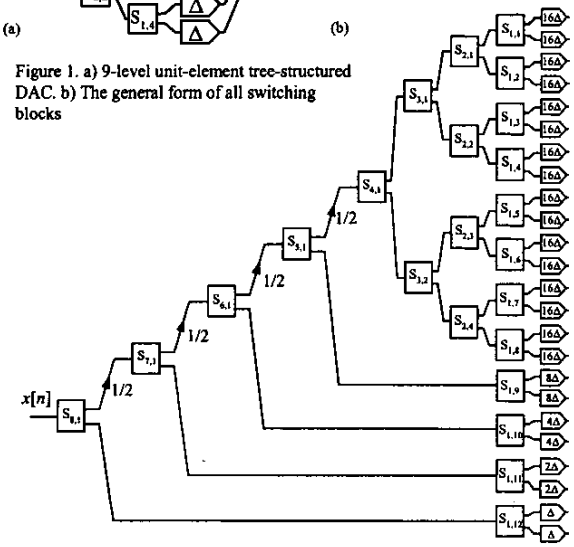

Figure 2. A 257-level partially-segmented tree-structured DAC.

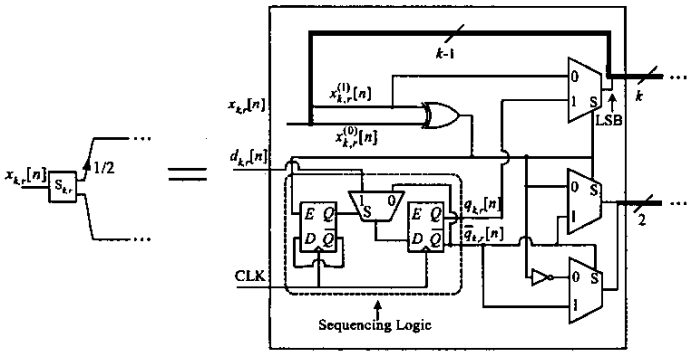

Figure 3. A hardware-efficient implementation of a switching block that quantizes to evens and divides the top output by two.

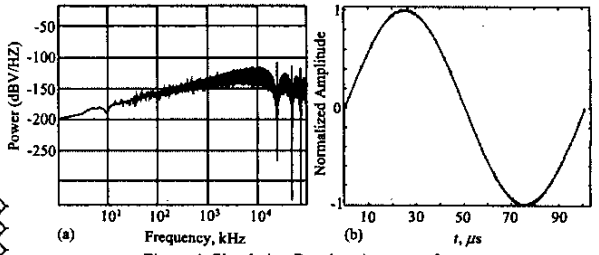

Figure 4. Simulation Results: a) output of system with input removed b) output time waveform.

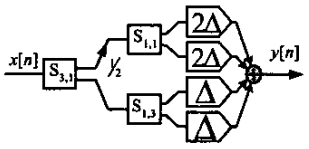

Figure 5. Fully-segmented 5-level tree-structured DAC.

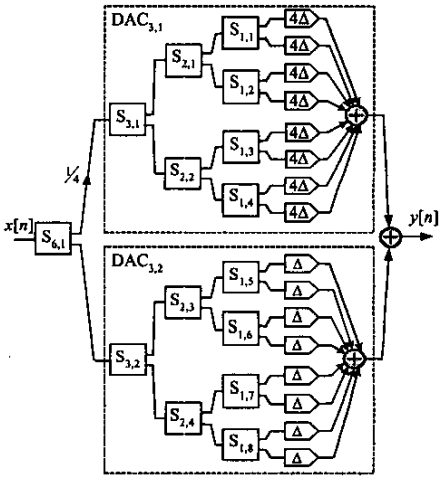

Figure 6. A partially-segmented 33-level tree-structured DAC.