DESIGN OF AN AUDIO DAC

Richard Schreier, Analog Devices

and

Gabor C. Temes, Oregon State University

EXAMPLE: AUDIO DAC [1]

Specifications:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Value</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input Sample Rate</td>
<td>$f_{s,in}$</td>
<td>44.1</td>
<td>kHz</td>
</tr>
<tr>
<td>Signal Bandwidth</td>
<td>$f_{b0}$</td>
<td>20</td>
<td>kHz</td>
</tr>
<tr>
<td>Output Signal-to-Noise Ratio</td>
<td>SNR</td>
<td>110</td>
<td>dB</td>
</tr>
<tr>
<td>Passband Flatness</td>
<td></td>
<td>0.1</td>
<td>dB</td>
</tr>
<tr>
<td>Image Attenuation</td>
<td></td>
<td>&gt;90</td>
<td>dB</td>
</tr>
<tr>
<td>Modulator Bandwidth</td>
<td>$f_{b} = 2 f_{s,in}$</td>
<td>88</td>
<td>kHz</td>
</tr>
<tr>
<td>Modulator Sample Rate</td>
<td>$f_{s} = 256 f_{s,in}$</td>
<td>11.29</td>
<td>MHz</td>
</tr>
<tr>
<td>Modulator OSR</td>
<td>OSR = $f_s/(2f_b)$</td>
<td>64</td>
<td></td>
</tr>
<tr>
<td>Number of Quantizer Steps</td>
<td>$M$</td>
<td>8</td>
<td></td>
</tr>
</tbody>
</table>
**DESIGN APPROACH**

System:

![Diagram of system: Digital Input, Interpolation Filter, Digital ΔΣ Modulator, DAC, Reconstruction Filter, Analog Output.]

To relax the analog filter specs, quadruple the passband width of the ΔΣ modulator to keep quantization noise away from $f_{80}$.

---

**MODULATOR DESIGN**

3rd-order, 9-level loop gives an SNR = 115 dB over a 0-88 kHz band.

![Diagram of 3rd-order, 9-level modulator.]

All NTF zeros at dc.
form = 'CIFB';
[a,g,b,c]=realizeNTF(NTF,form);
b(2:end) = 0;
ABCD=stuffABCD(a,g,b,c,form);
ABCDs = scaleABCD(ABCD,M+1);
[a,g,b,c] = mapABCD(ABCDs,form);
% scale for a(1) = 1/32
k = a(1)*32;
a(1) = a(1)/k;
b(1) = b(1)/k;
c(1) = c(1)*k;
% scale for c(1) = 0.5
k = c(1)/0.5;
c(1) = c(1)/k;
a(2) = a(2)/k;
c(2) = c(2)*k;
% scale for c(2) = 1
k = c(2)/1;
c(2) = c(2)/k;
a(3) = a(3)/k;
c(3) = c(3)*k;

---

08/02/04 temes@ece.orst.edu

---

QUANTIZED COEFFICIENTS

<table>
<thead>
<tr>
<th>Coefficient</th>
<th>Original Value</th>
<th>Transformed Value</th>
<th>Quantized Value</th>
<th>% Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>a₁ = b₁</td>
<td>0.0331</td>
<td>0.0312</td>
<td>1/32</td>
<td>0</td>
</tr>
<tr>
<td>a₂</td>
<td>0.0807</td>
<td>0.0617</td>
<td>1/16</td>
<td>1</td>
</tr>
<tr>
<td>a₃</td>
<td>0.1626</td>
<td>0.0909</td>
<td>1/16 + 1/32</td>
<td>3</td>
</tr>
<tr>
<td>c₁</td>
<td>0.6176</td>
<td>0.5000</td>
<td>1/2</td>
<td>0</td>
</tr>
<tr>
<td>c₂</td>
<td>1.3672</td>
<td>1.0000</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>c₃</td>
<td>10.4879</td>
<td>18.7539</td>
<td>16+2</td>
<td>-4</td>
</tr>
</tbody>
</table>
WORD LENGTH IN THE LOOP

For -130 dBFS truncation noise, the necessary word lengths can be found from the input-referred noise powers.

First integrator:
\[
\frac{(2^{-N_1})^2}{3a_1^2 OSR} < 10^{-13} (M^2/2)
\]
This leads to \(N_1 = 21\).

Second integrator:
\[
N_2 > -\log_2(Ma_1 c_1 \sqrt{0.45 \times 10^{-13} (OSR)^3}) = 16.2
\]
This gives \(N_2 = 17\).

Third integrator:
\[
N_3 > -\log_2(Ma_1 c_1 c_2 \sqrt{0.077 \times 10^{-13} (OSR)^3}) = 11.4
\]
This gives \(N_3 = 12\).

SIMULATED PSD

For a -1 dBFS input, the truncation noise over 0-20 kHz is -137 dBFS.
INTERPOLATION FILTER DESIGN

It reduces the unneeded spectral replicas between \( f_{b_0} \) and \( f_s - f_{b_0} \), to a level below the quantization noise \( Q_N \). Since the modulator STF also contributes attenuation, the filter needs to provide - \(|Q_N|^2(\text{dB}) + |\text{STF}|^2(\text{dB})\). Near \( f_B \), provide 90 dB.

![Graph showing STF and normalized frequency]

FILTER STRUCTURE

A cascade of interpolate – by – 2 sections:

![Filter diagram with sinc filters and normalized order]

Here, the \( \text{sinc}_N^k \) filter has the transfer function

\[
H(z) = \frac{(1 - z^{-N})^k}{N(1 - z^{-1})}
\]

where \( z \) refers to the output \( f_s \).
IMAGE GAIN

Defined as:

\[ G = \frac{H(e^{j2\pi f_1})}{H(e^{j2\pi f_{BO}})} \]

where \( f_1 = \left( \frac{f_s}{N} \right) - f_{BO} \). This is a function of \( N \) and \( OSR_{in} = \frac{f_s}{N} \cdot \frac{f_{BO}}{2} \).

---

SINC FILTER PERFORMANCE

---

08/02/04 temes@ece.orst.edu 10/20

08/02/04 temes@ece.orst.edu 11/20
SINC FILTER COMPLEXITY

<table>
<thead>
<tr>
<th>Parameter</th>
<th>sinc_2</th>
<th>sinc_2^5</th>
<th>sinc_3</th>
<th>sinc_2^2</th>
<th>sinc_4^1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output Sample Rate</td>
<td>8 f_{s,in}</td>
<td>16 f_{s,in}</td>
<td>32 f_{s,in}</td>
<td>64 f_{s,in}</td>
<td>256 f_{s,in}</td>
</tr>
<tr>
<td>Sinc order</td>
<td>7</td>
<td>5</td>
<td>3</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Number of Additions at the output sample rate (= Number of registers)</td>
<td>6</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Number of Additions at f_{s,in}</td>
<td>48</td>
<td>64</td>
<td>64</td>
<td>64</td>
<td>0</td>
</tr>
</tbody>
</table>

Polyphase implementation of sinc_2^3:

![Polyphase implementation diagram]

The modulator needs 2,048 additions at f_{s,in}!

SINC FILTERS' PASSBAND

![Sinc filters' passband diagram]

May be equalized in the first two stages.
SECOND FILTER STAGE

Saramaki halfband FIR filter [2][3]:

\[
[f1,f2,info] = \text{designHBF}(0.125,\text{undbv}(-70));
\]

\[
\text{figure}(1); \text{clf}
\]

\[
f = \text{linspace}(0,0.5,256);
\]

\[
\text{plot}(f*4, \text{dbv(frespHBF(f,f1,f2))},'b','Linewidth',1.5)
\]

\[
\text{figureMagic([0 2],0.25,2, [-140 10],10,2);
\]

\[
\text{printmif('HBF_freq', [5 2.5], 'Helvitical0')}
\]

\[
N = (2*\text{length}(f1)-1)*2*(2*\text{length}(f2)-1)+1;
\]

\[
y = \text{simulateHBF([1 zeros(1,N-1)],f1,f2);
\]

\[
\text{stem([0:N-1],y);
\]

\[
\text{figureMagic([0 N-1],5,2, [-0.2 0.5],0.1,1)
\]

\[
\text{printmif('HBF_imp', [5 2], 'Helvitical0')}
\]

SARAMAKI FILTER RESPONSE

82 dB image attenuation. It requires 44 additions, 50 registers. Halfband filter (symmetric response, fewer tap weights).
INPUT FILTER STAGE

- Passband 0 – 20 kHz, stopband suppression -90dB from $f_{s,in}/2$ to $2f_{s,in} - f_{B0}$.

- Cannot be halfband; stopband starts at $f_{s,in}/2$.

- Last designed, so it can equalize the droop of the following stages.

- Usually, the stopband only starts at $f_{s,in} - f_{B0}$ (here, 24.1 kHz instead of 22.05 kHz). Much less complex!

- MATLAB's remez function can be used to design it: 186th-order FIR, 445 additions, 186 registers, 60% of filter power dissipation.

FREQUENCY RESPONSES

Overall response:

Passband response:
DAC AND RECONSTRUCTION FILTER

Transfer function:

\[ \frac{V_{out}(s)}{I_{in}(s)} = \frac{2R}{\left(\frac{s}{\omega_0}\right)^2 + \left(\frac{s}{\omega_0Q}\right) + 1} \]

where \( Q = \sqrt[1/2]{\frac{C_1}{2C_2}} \)

\( \omega_0 = \frac{1}{\sqrt[1/2]{C_1C_2R}}. \)

FILTER DESIGN

- For -90 dBFS / 22 kHz noise PSD, a Butterworth response with \( f_{3db} = 80 \text{ kHz} \) may be used. Then, \( Q = 1/\sqrt{2} \) and \( C_1 = C_2 = C \), \( RC \approx 1.41 \times 10^{-6} \text{ s} \).
- Output noise PSD is 16 kTR at low frequencies. For SNR = 116 dB, \( R = 1 \text{ k}\Omega \) and \( C = 1.41 \text{ nF} \) may be used.
- Then for a FS \( V_{out} = 0.7 \ V_p, i_{in} = 0.35 \ mA_p \). 8 unit elements, each 39 \( \mu A \).
REFERENCES

