The Virtual Buchla 259 Wavefolder

Application of Virtual Analog Instrument

Anna Obara
Aalborg University Copenhagen
Sound and Music Computing, 7th Semester
Saturday 8th February, 2025

***Abstract*—This project explores the digital modeling and real-time implementation of the Buchla 259 Wavefolder, a pivotal component in West Coast synthesis, known for its nonlinear wavefolding and timbral richness. Employing MATLAB for prototyping, the work emphasizes preserving the spectral characteristics of the original analog design while mitigating digital artifacts like aliasing. The methodology encompasses a comprehensive circuit analysis, mathematical modeling of non-linear folding stages, and the application of anti-aliasing polyBLAMP technique. The resulting MATLAB prototype is evaluated against theoretical benchmarks and analog simulations, focusing on harmonic accuracy and aliasing suppression. The simulation was fully performed based on Esqueda's work [1].**

I. Introduction

The objective of this project is to simulate the nonlinear waveshaping behavior of the original analog circuit while minimizing aliasing artifacts. It follows a virtual analog modeling approach, employing memoryless nonlinear mappings to replicate the input-output characteristics. Since nonlinear distortion introduces aliasing, a polyBLAMP method is applied to suppress unwanted artifacts while maintaining computational efficiency. MATLAB is used to develop and test the wavefolder model, ensuring that it retains the characteristics of the original analog design while operating efficiently in a digital audio environment. Finally, the results are evaluated to compare the effectiveness of the polyBLAMP method in reducing aliasing while preserving the desired timbral characteristics of the wavefolded signal.

II. Comparative Analysis

In his original study, Esqueda conducted an analysis of the Buchla circuit. Based on this, he derived a static, memoryless function to establish the mapping between the input and output signals. During the implementation process, however, the summing amplifier stage was omitted. Consequently, the behavior of each individual cell was characterized solely by its threshold and weight, the latter encapsulating the cell's polarity within the summation process.

$$\text{thresholds} = \frac{R(k,2)}{R(k,1)} \tag{1}$$

Resistor-weighted voltage summation have been replaced by precomputed scalar values. These are precomputed constants that approximates the functional characteristics of an analog circuit. In digital equivalent, the multiplication coefficient is essential because it preserves the dynamic range and it contributes to the harmonic complexity of the wavefolded signal.

$$\text{weighting factor} = \frac{RF1}{Rk,3} \tag{2}$$

for lower amplifier, and

$$\text{weighting factor} = \frac{RF2}{Rk,3} \tag{3}$$

for upper amplifier, which is also the output of the circuit.

In analog circuit, inverting amplifiers perform two primary functions. Firstly, they integrate multiple folding stages while maintaining precise gain factors to ensure accurate signal summation. Secondly, they employ negative feedback mechanisms to regulate and shape the voltage response characteristics of each folding cell, enhancing stability and linearity in signal processing.

$$V_{\text{out}} = -\left(\frac{R_2}{R_1}\right) V_{\text{in}} \tag{4}$$

In the digital implementation, direct inverting amplification is not required; instead, an inverse clipping process is introduced. The inverse clipper replicates the nonlinear folding behavior within a discrete-time system. It preserves signal peaks while selectively attenuating the mid-range amplitude, ensuring a consistent nonlinear response.

The final stage of the process is characterized by low-pass filtering effects introduced by the summing amplifier, along with the intrinsic bandwidth limitations of the analog components, which collectively result in the attenuation of high-frequency signal components. In contrast, the digital model employs polyBLAMP correction, which replaces sharp signal transitions with a smooth, band-limited representation, effectively mitigating aliasing artifacts and preserving spectral integrity in virtual implementation.

III. MATLAB Implementation

The following pseudo-code implements a wavefolding nonlinearity based on the mathematical formulation.

A. Input Signal Generation

The first snippet presents the procedure for generating the input signal, which will subsequently be processed through the wavefolding function.

Algorithm 1 Input signal generation

procedure WAVEFOLDERSIMULATION(A, f_0, f_s, V_s, R)
  $T \leftarrow 1/f_s$ //Sampling period
  $T_{\text{total}} \leftarrow 2/f_0$ //Total simulation time for 2 cycles
  $t \leftarrow [0, T, \ldots, T_{\text{total}}]$ //Time vector
  $V_{\text{in}} \leftarrow A \cdot \sin(2\pi f_0 t)$ //Generate sinusoidal waveform
end procedure=0

The resistor values are provided as input parameters within the initial algorithmic implementation, structured in accordance with the dataset presented in Esqueda's research.

B. Clipping Points Calculation

Algorithm 2 Computing clipping thresholds and points

procedure WAVEFOLDERSIMULATION(A, f_0, f_s, V_s, R)
  $thresholds \leftarrow (R[:,1]/R[:,2]) \cdot V_s$
  $t_1, t_2, t_3, t_4 \leftarrow$ ComputeClippingPoints(R, A, f_0, V_s)
end procedure=0

This calculation determines the voltage level `(tresholds)` at which the signal is clipped for each cell. It uses the ratio of the resistors `R[:,1]` and `R[:,2]` multiplied by the supply voltage `Vs`. Clipping points are the exact time points where the input signal reaches the clipping thresholds. These points are essential for identifying waveform discontinuities and applying anti-aliasing corrections. The parameters corresponding to individual folding cells are presented in the TABLE I.

TABLE I: Clipping points (in seconds)

Folding Cell	t_1	t_2	t_3	t_4
1	0.000191	0.004809	0.005191	0.009809
2	0.001022	0.003978	0.006022	0.008978
3	NaN	NaN	NaN	NaN
4	0.000586	0.004414	0.005586	0.009414
5	0.001519	0.003481	0.006519	0.008481

C. Fractional Delay

The subsequent step involves the calculation of the fractional delay (d). It represents the initial phase in implementing the polyBLAMP correction residual function. The implementation described in the referenced paper outlines an analytical method for computing the clipping points based on a continuous-time sinusoidal input. This approach enables the precise determination of the exact time t_k at which each clipping point occurs within the continuous sinusoidal waveform. The clipping point in time is located between two discrete samples.

Those three steps are critical for ensuring that the wavefolding process avoids aliasing artifacts, which can occur when discontinuities in the waveform introduce high-frequency components that exceed the Nyquist frequency $(f_s/2)$. Here, scaling factor is calculated, which determines the amplitude of the correction based on the discontinuity magnitude. Then, the

Algorithm 3 Anti-Aliasing factors

procedure WAVEFOLDERSIMULATION(A, f_0, f_s, V_s, R)
  $\mu \leftarrow$ ComputePolyBLAMPScalingFactor(A, f_0, t_1, f_s)
  $n_k, d_k \leftarrow$ ComputeSampleIndices(t_1, t_2, t_3, t_4, T)
  $R_{\text{pre}}, R_{\text{post}} \leftarrow$ ComputePolyBLAMPResiduals(d_k)
end procedure=0

applied formula determines d as the proportional progression of the signal between two consecutive samples, n and $n+1$, at which the threshold is surpassed. The fractional delay is determined by scaling to the sampling grid:

$$d_k = \frac{t_k}{T} - n_k \tag{5}$$

where n_k = nearest sample index before t_k

`nk = floor([t1, t2, t3, t4] / T);` and `floor` function rounds each element of A down to the nearest integer (greatest integer less than or equal to A). It finds the closest integer sample index. `dk = ([t1, t2, t3, t4] / T) - nk;` gives the fractional position of the clipping event between two samples. `R_pre` controls the amount of correction applied before the clipping point, while `R_post` controls the amount of correction applied after the clipping point. TABLE II shows the calculation results for five folding cells.

TABLE II: Fractional delays for 5 folding cells

FC	t_1, n	t_2, n	t_3, n	t_4, n
1	(0.3195, 6)	(0.8827, 191)	(0.5218, 204)	(0.0850, 390)
2	(0.9609, 32)	(0.2414, 165)	(0.1631, 231)	(0.4436, 363)
3	(0.1295, 72)	(0.0728, 126)	(0.3317, 270)	(0.2750, 324)
4	(0.2230, 19)	(0.9793, 178)	(0.4252, 217)	(0.1815, 377)
5	(0.1761, 47)	(0.0261, 151)	(0.3784, 245)	(0.2283, 349)

Fractional delay, as a measure of the position of a clipping point is a unitless quantity, and is presented as the first value in each column. The sample index n represents the discrete time step within a digital signal at which a specific event, such as a threshold crossing, is detected. This index is indicated as the second value in each column. A positive fractional delay $d > 0$ indicates that an event occurs after the nearest discrete sampling instant at nT. Specifically, a fractional delay of $d = 0.4428$ implies that the event takes place approximately $\sim 44,28\%$ of the interval between the current sample at nT and the subsequent sample at $(n+1)T$, where T denotes the sampling period. Formally, when the fractional delay is $d = 0$, the event aligns precisely with a discrete sample point. Conversely, when $d = 1$, the event coincides exactly with the subsequent sample point in the sequence.

D. Inverse clipping

The process of implementation is applied symmetrically, meaning both positive and negative thresholds are enforced equally. The resulting clipped signal is intended for further processing, where anti-aliasing corrections will be ap-

Algorithm 4 Apply clipping

procedure WAVEFOLDERSIMULATION(A, f_0, f_s, V_s, R)
  $V_{\text{clip}} \leftarrow$ InitializeMatrix$(5, \text{length}(V_{\text{in}}))$
  for $k \leftarrow 1$ to 5 **do**
    for $n \leftarrow 1$ to length(V_{in}) **do**
      if $|V_{\text{in}}[n]| > thresholds[k]$ **then**
        $V_{\text{clip}}[k, n] \leftarrow V_{\text{in}}[n]$
      else
        $V_{\text{clip}}[k, n] \leftarrow \text{sign}(V_{\text{in}}[n]) \cdot thresholds[k]$
      end if
    end for
  end for
end procedure=0

plied to smooth out discontinuities that occur at the clipping points, ensuring a more refined and natural output. If the signal exceeds the threshold, it remains unchanged. `abs(Vin[n]) > thresholds[k]` determines if the signal exceeds the threshold and if it does, the signal is left unchanged because it is beyond the fold point. `sign(Vin[n])` extracts the sign (+ or -) of the signal. Ultimately, the threshold value is multiplied by this sign to ensure that the clipped signal retains the correct polarity.

E. Residual corrections

The first residual represents the correction values applied prior to the clipping point, corresponding to the interval ($[-T, 0]$). Conversely, the second residual contains values that represent the correction applied after the clipping point, within the interval ($[0, T]$). Small values (~ 0.00003) appear at fractional delays close to 0 or 1, where the cubic function in the polyBLAMP residual is very small. Larger values on the other hand (~ 0.16) appear at fractional delays closer to 0.5, where the polyBLAMP function "peaks". It means that the magnitude of the correction is largest when the fractional delay $d = 0.5$.

For the second residual, the pattern is approximately a mirrored version of the first residual. Specifically, residual values are close to 0 for very small fractional delays d. In the first residual, the most significant correction occurs at small d values, whereas in the second residual, the largest correction is observed at larger d values.

`for k = 1:5` ensures that each folding stage is processed independently.

`for n = 1:4` this loop verifies that all clipping points are handled.

`clip_s = nk[k, n] + 1` shifts the index to align with MATLAB indexing. `nk[k, n]` is the integer sample index where the signal crosses a clipping threshold.

`Vin[clip_s]<Vin[clip_s+1]` checks whether the transition exhibits an upward trend. `else` means the signal is falling (downward slope). If the condition is met and the waveform is rising, it could be either positive-going transition, from negative to positive (`Vin[clip_s+1]>0`) or negative-going transition from positive to negative (`Vin[clip_s+1]<0`). The correction is applied in reverse order, with less correction on the future sample and more on the current sample. Analogous conditions must be met when transition exhibits an downward trend (`Vin[clip_s]>Vin[clip_s+1]`).

Algorithm 5 PolyBLAMP Correction at Detected Discontinuities

procedure WAVEFOLDERSIMULATION(A, f_0, f_s, V_s, R)
  for each folding cell $k \in \{1, 2, 3, 4, 5\}$ **do**
    for each clipping event $n \in \{1, 2, 3, 4\}$ **do**
      $clip\_s \leftarrow nk[k, n] + 1$ {Determine sample index}
      if $Vin[clip\_s] < Vin[clip\_s + 1]$ **then** {Upward transition}
        if $Vin[clip\_s + 1] > 0$ **then** {Positive-going transition}
          $Vclip[k, clip\_s+1] \leftarrow Vclip[k, clip\_s+1] + R_{post}[k] \cdot \mu[k]$
          $Vclip[k, clip\_s] \leftarrow Vclip[k, clip\_s] + R_{pre}[k] \cdot \mu[k]$
        else{Negative-going transition}
          $Vclip[k, clip\_s+1] \leftarrow Vclip[k, clip\_s+1] + R_{pre}[k] \cdot \mu[k]$
          $Vclip[k, clip\_s] \leftarrow Vclip[k, clip\_s] + R_{post}[k] \cdot \mu[k]$
        end if
      else{Downward transition}
        if $Vin[clip\_s] > 0$ **then** {Positive-to-negative transition}
          $Vclip[k, clip\_s+1] \leftarrow Vclip[k, clip\_s+1] - R_{pre}[k] \cdot \mu[k]$
          $Vclip[k, clip\_s] \leftarrow Vclip[k, clip\_s] - R_{post}[k] \cdot \mu[k]$
        else{Negative-to-positive transition}
          $Vclip[k, clip\_s+1] \leftarrow Vclip[k, clip\_s+1] - R_{post}[k] \cdot \mu[k]$
          $Vclip[k, clip\_s] \leftarrow Vclip[k, clip\_s] - R_{pre}[k] \cdot \mu[k]$
        end if
      end if
    end for
  end for
end procedure=0

F. Folded output

This step processes the clipped signals (`Vclip`) to produce the folded output for each folding cell. It combines the clipped signals with scaling factors derived from the resistor network. The resistors in each folding cell determine how much the clipped signal is scaled before being included in the output. `denom` accounts for the combined effects of the three resistors in each folding cell, while `factor` contributes to the numerator. The folded output for each cell is the scaled and corrected clipped signal.

Algorithm 6 Anti-Aliasing factors

procedure WAVEFOLDERSIMULATION(A, f_0, f_s, V_s, R)
  for $k \leftarrow 1$ to 5 **do**
    for $n \leftarrow 1$ to length(V_{clip}) **do**
      $denom \leftarrow (R[k,1] \cdot R[k,3]) + (R[k,2] \cdot R[k,3]) + (R[k,1] \cdot R[k,2])$
      $factor \leftarrow (R[k,2] \cdot R[k,3])/denom$
      $V_{\text{out}}[k,n] \leftarrow factor \cdot (V_{\text{clip}}[k,n] - \text{sign}(V_{\text{clip}}[k,n]) \cdot (V_s \cdot R[k,1]/R[k,2]))$
    end for
  end for
end procedure=0

G. Global summing stage

Algorithm 7 Global summing stage

procedure COMPUTEGLOBALOUTPUT($V_{\text{out}}, V_{\text{in}}$)
  $V_{\text{global}} \leftarrow$ InitializeVector(length($V_{\text{out}}[1,:]$))
  for $n \leftarrow 1$ to length(V_{global}) **do**
    $V_{\text{global}}[n] \leftarrow (-12.000 \cdot V_{\text{out}}[1,n])$
      $+(-27.777 \cdot V_{\text{out}}[2,n])$
      $+(-21.428 \cdot V_{\text{out}}[3,n])$
      $+(17.647 \cdot V_{\text{out}}[4,n])$
      $+(36.363 \cdot V_{\text{out}}[5,n])$
      $+(5.000 \cdot V_{\text{in}}[n])$
  end for
end procedure=0

Each folding cell's contribution is weighted by a unique factor. Negative weights invert the signal, which can enhance harmonic cancellation or emphasize specific components. The input signal is included with its own weight (e.g. 5.000) to ensure that the final output has some resemblance to the original signal. By modulating the weighting coefficients applied to each stage, the harmonic balance can be adjusted dynamically, enabling fine control over the spectral distribution and tonal coloration of the processed signal.

IV. RESULTS

The digital implementation of the Buchla 259 wavefolder was evaluated based on its ability to replicate the nonlinear waveshaping characteristics of the original circuit while minimizing aliasing artifacts. The model was tested using sinusoidal and complex input signals to observe its harmonic generation and spectral behavior. To address aliasing artefacts, an oversampling factor of 8× was applied in conjunction with the bandlimited ramp (BLAMP) method. Spectral analysis (Figure 2) confirmed that aliasing components were substantially reduced, resulting in cleaner harmonic content. The digital wavefolder successfully preserved the expected harmonic structures and dynamic timbral variations associated with analog wavefolding. Figure 1 illustrates the dynamic nature of the wavefolder, where small changes in the input amplitude lead to significant changes in the output waveform. This behavior is central to the wavefolder's ability to generate diverse timbral variations in a synthesis context. The final implementation achieved a balance between computational efficiency and accuracy, making it suitable for real-time synthesis applications.

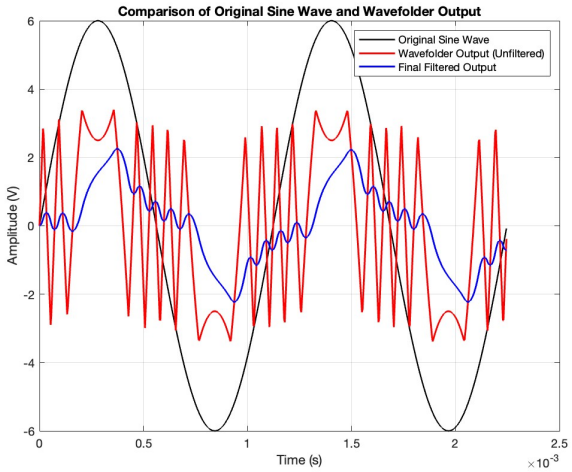

Fig. 1: Time-Domain Waveform Comparison of Original and Processed Signals

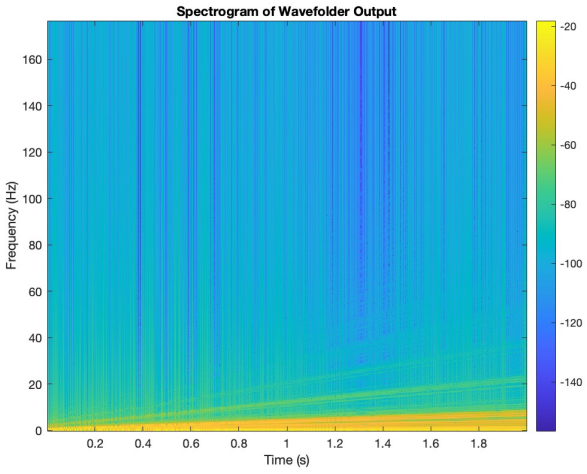

Fig. 2: Spectrogram

V. CONCLUSIONS

This study detailed the digital implementation of the Buchla 259 wavefolder, a nonlinear waveshaping circuit used in West Coast synthesis. The proposed model replicates the unique folding behavior of the original analog circuit, capturing its distinctive timbral transformations.

The results confirmed that this approach effectively suppresses spectral artifacts while maintaining the harmonic complexity of the wavefolder. Compared to basic waveshaping functions, the implemented model offers a more faithful reproduction of the Buchla 259's sonic characteristics.

The developed wavefolder can be integrated into virtual modular synthesizers, software-based synthesis environments, and real-time digital audio applications. Future improvements may focus on optimizing computational efficiency, extending modulation capabilities, and exploring alternative antialiasing techniques to further refine the model's performance.

REFERENCES

[1] F. Esqueda, H. Pöntynen, V. Välimäki, and J. D. Parker, "Virtual analog buchla 259 wavefolder," in *Proceedings of the 20th International Conference on Digital Audio Effects (DAFx-17)*, Edinburgh, UK, September 2017, pp. 1–8.