HABIB UNIVERSITY

. **Digital Signal Processing** **Instructor: Dr. Shafayat Abrar**

. **Lab 05: Sigma-Delta Modulation** **Demonstrator: Ahmad Bilal**

1 Objectives

- Understand the basics of $\Sigma\Delta$ modulation.
- Implement a first-order $\Sigma\Delta$ modulator in MATLAB.
- Analyze the performance of the modulator using time and frequency domain representations.

2 Pre-requisites

- Basic knowledge of digital signal processing.
- Familiarity with MATLAB programming.

3 Introduction

In Lab 02, we discussed line coding (LC) or pulse code modulation (PCM) in digital pulse communication. PCM, while accurate, involves complex encoding and decoding processes, leading to high implementation costs in hardware and computational resources. A low-cost alternative to PCM is DM offers a more efficient solution by oversampling the modulating signal, simplifying the encoding process, and reducing hardware complexity. In PCM, various operations such as encoding, decoding, and quantizing contribute to the formation of a complex circuit, making practical implementation costly. In contrast, the Sigma-Delta Modulation (SDM) streamlines these processes, resulting in a more economical and efficient digital communication technique.

Sigma-Delta Modulation (SDM) is a widely used technique in analog-to-digital (A/D) and digital-to-analog (D/A) conversion, particularly in high-resolution audio and sensor applications. It employs oversampling and noise shaping to achieve high-resolution digital representation using low-bit quantization, typically a single bit. This technique is highly effective in reducing quantization noise within the signal bandwidth while shifting it to higher frequencies where it can be easily filtered out.

4 Quantization and Sampling-Hold

The fundamental principle of Sigma-Delta modulation is based on the combination of oversampling, quantization, and feedback. The process involves:

- **Sampling-Hold**: The input signal is first sampled at a much higher rate than the Nyquist rate to provide a detailed representation of the signal.

- **Quantization**: The sampled signal is quantized using a low-bit quantizer, often a single-bit quantizer that represents the signal as either +1 or -1.

This coarse quantization introduces quantization noise, which is mitigated using the feedback mechanism in the Sigma-Delta modulator.

5 Feedback Mechanism for Prediction and Encoding

The feedback loop in Sigma-Delta modulation plays a crucial role in predicting and encoding the signal effectively. It consists of:

- **Integrator**: The difference between the input signal and the previous quantized output is integrated over time, allowing the system to track slow variations in the signal.
- **Predictive Filtering**: By feeding back the quantized signal and subtracting it from the input, the system minimizes the difference between the actual and predicted values, ensuring better representation over time.
- **Noise Shaping**: The quantization noise is shaped and pushed to higher frequencies, reducing its impact on the signal within the band of interest.

This predictive encoding ensures that the output bitstream efficiently captures variations in the input signal while maintaining a simple one-bit representation.

6 Recovery Mechanism at the Receiver

At the receiver, the original signal is reconstructed using a low-pass filter to remove high-frequency quantization noise. The steps include:

- **Digital Filtering**: A decimation filter is applied to downsample the bitstream and reconstruct the original signal with higher resolution.
- **Low-Pass Filtering**: The noise components that were shaped to higher frequencies are filtered out, leaving behind a clean representation of the input signal.
- **Reconstruction**: The final signal is interpolated and converted back to an analog form (in D/A conversion scenarios) for use in downstream applications.

7 Step-By-Step Implementation Details

1. **Oversampling:** The first step in the $\Sigma\Delta$ modulation process is **oversampling**. Unlike conventional sampling with B-bit quantization (as used in Lab 04), where the sampling frequency is given by

$$f_S = K_1 f_{\max}, \quad K_1 \geq 2,$$

the $\Sigma\Delta$ modulator increases the sampling rate further. Specifically, the input signal $x(t)$ is sampled at a frequency

$$f_{S,\Sigma\Delta} = K_2 K_1 f_{\max}.$$

The choice of K_2 is discussed in point 5.

2. **Error Quantization Instead of Direct Quantization:** Unlike conventional quantization, where the signal samples are quantized directly, a $\Sigma\Delta$ modulator applies quantization to the **prediction error**. The error at time index n is defined as the difference between the current discrete-time sample $x[n]$ and the previously predicted quantized sample $x_Q[n-1]$:

$$e[n] = x[n] - x_Q[n-1].$$

This approach leverages **feedback** to track changes in the signal rather than the absolute values.

3. **One-Bit Quantization of the Error:** The $\Sigma\Delta$ modulator performs **one-bit quantization** of the error $e[n]$ into one of two possible values: $+\Delta$ or $-\Delta$. The quantized error signal is given by:

$$e_Q[n] = \Delta \operatorname{sign}(e[n]),$$

where

$$e_Q[n] = \begin{cases} +\Delta, & \text{if } e[n] > 0, \\ -\Delta, & \text{if } e[n] < 0. \end{cases}$$

This **binary quantization** simplifies the encoding and transmission process.

4. **Encoding and Transmission:** Since the quantized error takes only two possible values, the $\Sigma\Delta$ modulator transmits only **one-bit information per sample**. The encoding follows this mapping:

$$+\Delta \to \mathbf{1}, \quad -\Delta \to \mathbf{0}.$$

In this lab, we may omit the explicit encoding step and focus on the transmission of the quantized symbols $\pm\Delta$. At the receiver, the goal will be to reconstruct the original analog signal from these quantized symbols.

5. **Comparison with Conventional Quantization:** To compare the $\Sigma\Delta$ modulator with **conventional sampling and B-bit quantization**, we set $K_2 = B$. This choice ensures that, rather than transmitting B bits per quantized sample, the $\Sigma\Delta$ modulator oversamples the input signal at a rate Bf_S and transmits only **one-bit per sample**.

6. **Recursive Signal Reconstruction:** At the receiver, the **predicted quantized sample** $x_Q[n]$ is recursively computed from the previous quantized sample $x_Q[n-1]$ and the received quantized error $e_Q[n]$:

$$x_Q[n] = x_Q[n-1] + e_Q[n].$$

This **integration step** is crucial for signal reconstruction. The initial value of $x_Q[n]$ may be taken as zero.

7. **Block Diagram Representation:** The complete $\Sigma\Delta$ modulation process, including **oversampling, error computation, quantization, feedback, and reconstruction**, can be visualized through a block diagram, as shown below:

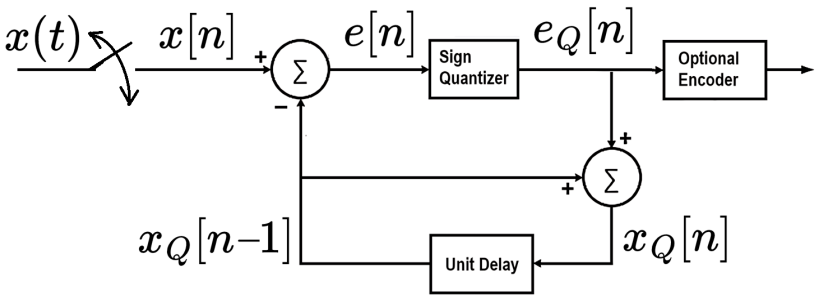

Figure 1: Block diagram illustrating the $\Sigma\Delta$ modulation process, showing the sampler, subtractor (error computation), quantizer, and the feedback loop.

8. **Error Accumulation at the Receiver:** At the receiver, the task is to accumulate the received quantized error symbols to reconstruct the signal. This accumulation can be implemented using a **first-order delayed feedback system**, as shown in the block diagram below:

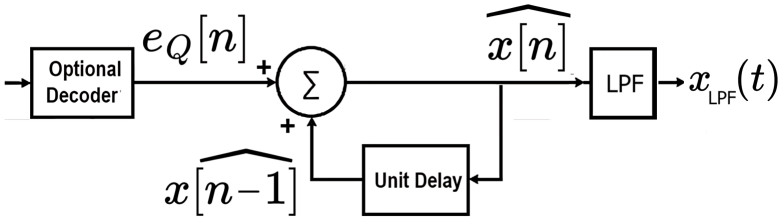

Figure 2: Block diagram illustrating the $\Sigma\Delta$ demodulation process, showing the accumulator and lowpass filter.

9. **Mathematical Expression for Reconstructed Signal:** The reconstructed signal $\widehat{x[n]}$, obtained from the accumulation of quantized error values, can be expressed as:

$$\widehat{x[n]} = \sum_{m=-\infty}^{n} e_Q[m] = \widehat{x[n-1]} + e_Q[n].$$

   In MATLAB, this can be efficiently implemented using the **cumsum** (cumulative sum) command. By applying **cumsum** to the entire sequence of received signal $e_Q[n]$, we can generate a plot of $\widehat{x[n]}$ or $\widehat{x(t)}$ over the given time scale.

10. **Low-Pass Filtering for Signal Reconstruction:** The final step is to use an appropriate **low-pass filter** to remove high-frequency components from $\widehat{x(t)}$. This filtering process smooths the reconstructed signal, yielding an estimate $x_{\text{LPF}}(t)$ that closely resembles the original sampled signal $x(t)$. The choice of filter parameters is crucial to achieving a high-fidelity reconstruction. The signals at the receiver side are shown in Figure below:

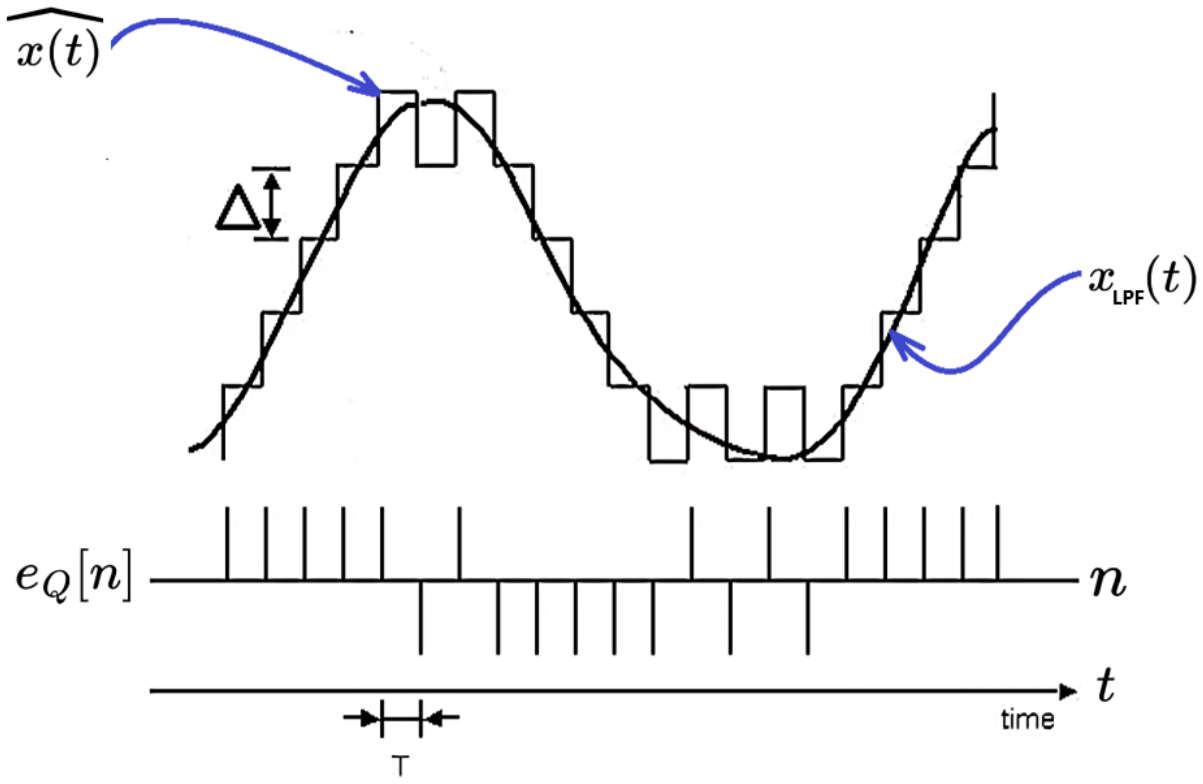

Figure 3: The signals at the receiver side.

Activity 1: Consider a sampling signal

$$x(t) = A_1 \cos(2\pi f_1 t) + A_3 \cos(2\pi f_3 t)$$

Assume that

$$A_1 = 1$$
$$A_3 = 1$$
$$f_1 = 1$$
$$f_3 = 3f_1$$
$$K_2 = B = 8$$
$$K_1 \geq 5$$
$$f_S = K_1 K_2 f_3$$

Now implement Sigma-Delta modulator as shown in Fig. 1. Choose an appropriate value of Δ to generate $e_Q[n]$. To obtain the quantized error signal, $e_Q[n]$, for the sampling signal $x(t)$, you need to

simulate the following equations in a loop for every sample of $x(t)$

$$e[n] = x[n] - x_Q[n-1]$$
$$e_Q[n] = \Delta\,\text{sign}(e[n])$$
$$x_Q[n] = x_Q[n-1] + e_Q[n]$$

If n is starting from 1, then the required initial value $x_Q[0]$ may be set to 0. Note use MATLAB **`stem`** (without marker symbol) to plot $e_Q[n]$.

If everything is done correctly, then you obtain the following plots of $x(t)$ and $e_Q[n]$:

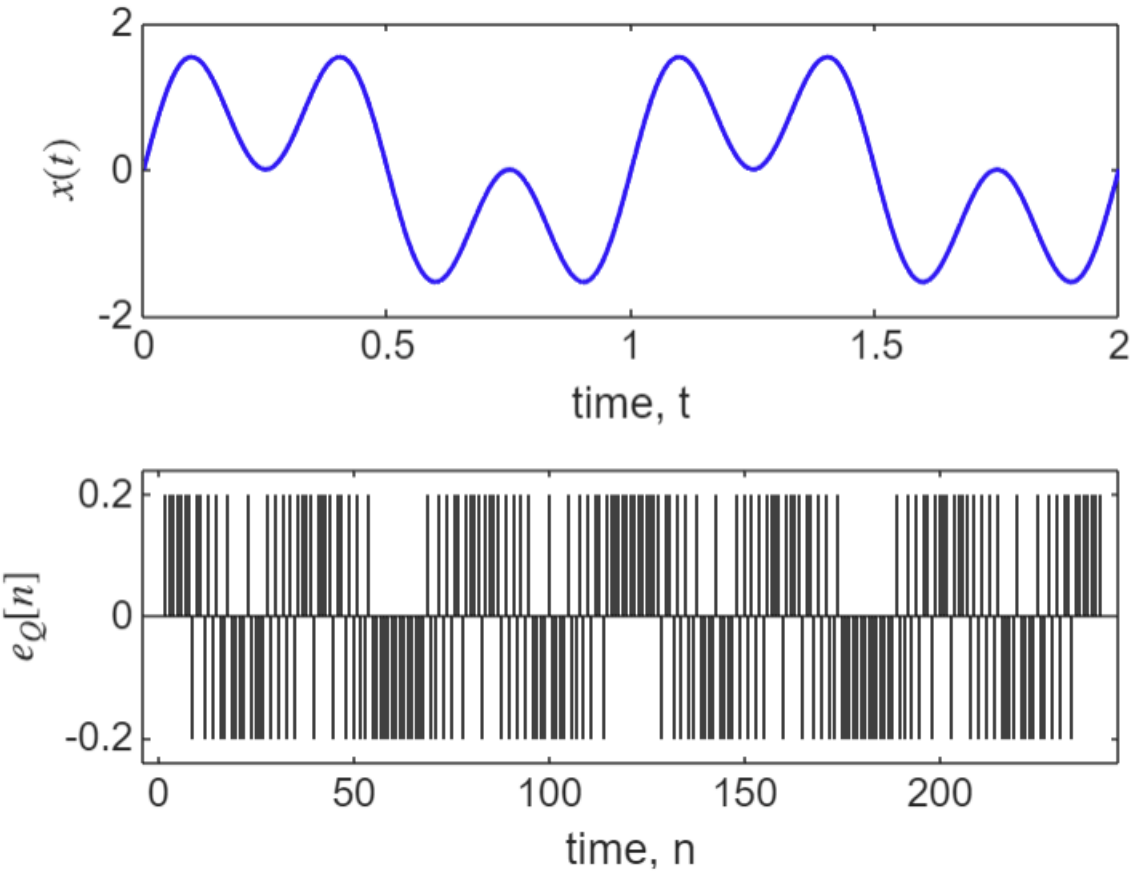

Figure 4: The sampling signal and the transmitted quantized error signal where we assumed $\Delta = 0.2$.

Activity 2: Now implement the receiver end of Sigma-Delta modulator as shown in Fig. 2. As mentioned earlier, you may use the MATLAB cumsum command to obtain $\widehat{x[n]}$ directly from $e_Q[n]$. Then apply an appropriate lowpass filter to obtain the finally recover the original signal $x(t)$.

Hints for Activity 2: To apply a low-pass filter to the signal $\widehat{x[n]}$, you need to insert $M-1$ zeros between each pair of consecutive levels of $\widehat{x[n]}$. This zero-padding process facilitates interpolation by the low-pass filter, allowing it to generate a smooth analog signal from the quantized data. Consider $M=5$, and obtain an $M-1$ zero-padded signal $\widehat{x_M[n]}$. Next you may design a lowpass filter using the sinc function as follows:

$$h(t) = \texttt{sinc([-p*M:p*M]*fs*dt/p)/p;}$$
$$x_{\text{LPF}}(t) = \texttt{filter(}h(t), 1, \widehat{x_M[n]}\texttt{);}$$

where `dt` is the smallest time step $T_S/M = 1/(Mf_S)$ and `fs` is the sampling frequency f_S. Observe and note down the performance of the lowpass filter for $p = 4, 5, 6, 7, \cdots$.

If everything goes well, then you obtain the following plots of $\widehat{x(t)}$ and $x_{\text{LPF}}(t)$:

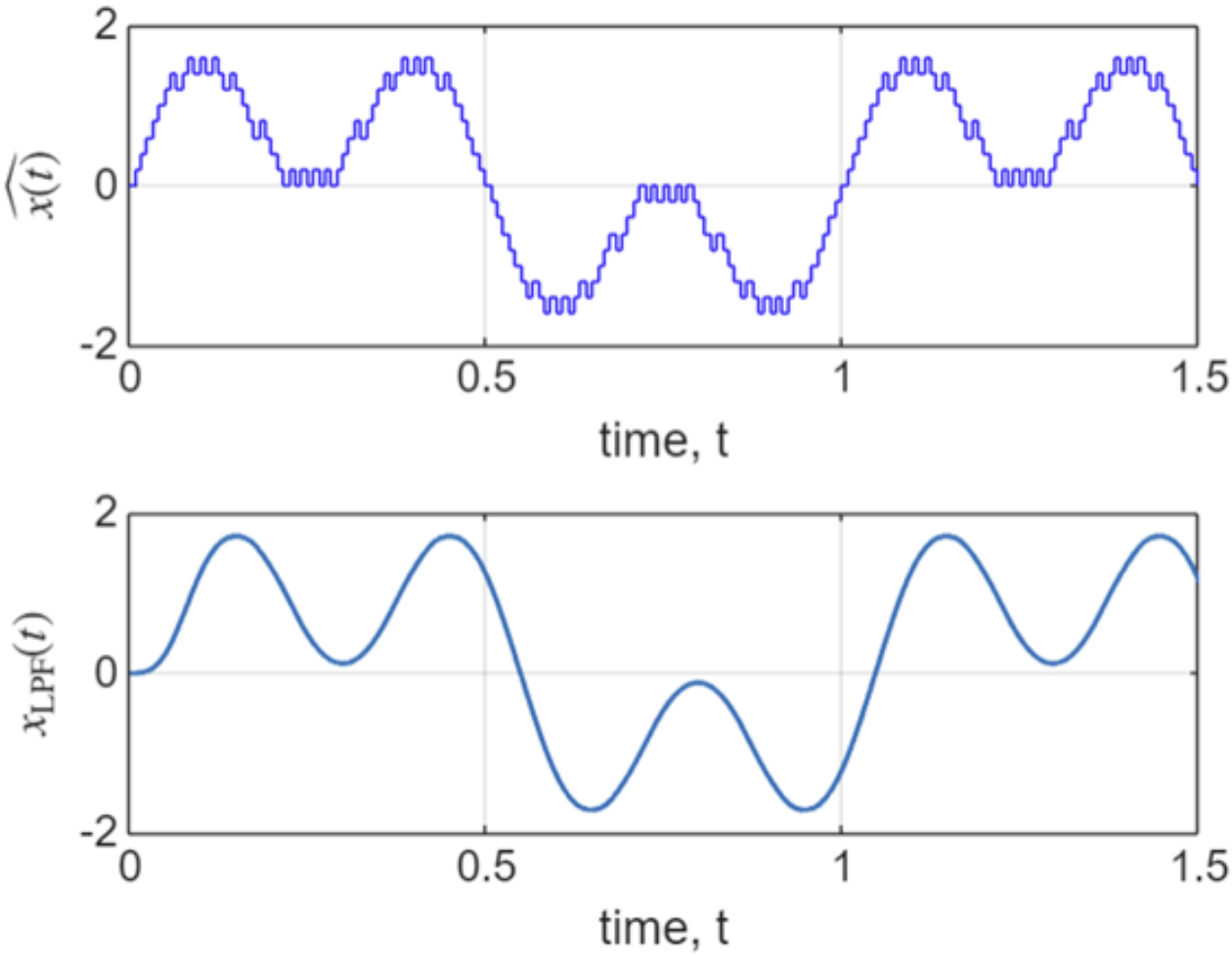

Figure 5: The input and output of the lowpass filter at the receiver.

Activity 3: Apply sigma-delta modulation on an audio signal. Provide all steps of your work.