Project Specification Report

Line Encoding Visualizer

Objective

The objective of this project is to visualize various **Digital-to- Digital** and **Analog-to-Digital** encoding schemes used in data communication systems. The tool provides an interactive way to understand how binary and analog data are encoded for transmission, displaying each scheme's waveform using graphical plots.

Programming Language and Libraries Used

- **Language:** Python 3.x
- **Libraries:**
  - **NumPy:** Used for numerical operations and efficient list handling.
  - **Matplotlib:** Used for plotting time–amplitude waveforms of the encoded signals.

Assumptions Considered

- Each bit duration is assumed to be 1 unit of time.
- Positive Logic is followed initially.
- Logical '1' and '0' are represented by amplitude levels +1 and –1 depending on the encoding scheme.
- Sampling values in PCM are within a fixed amplitude range (–3 to +4 for PCM).
- Transitions in Manchester and Differential Manchester occur at the middle of each bit period.
- Ideal signal transmission is assumed (no noise or distortion).

Methodology

1. The user selects whether the input signal is **Digital** or **Analog**.

2. For digital input, the program converts a given binary sequence into one of the encoding schemes: **NRZ-L, NRZ-I, Manchester, Differential Manchester, or AMI**.

3. For analog input, two conversion schemes are supported:

   - **Pulse Code Modulation (PCM)** – performs quantization, binary encoding, and line encoding.
   - **Delta Modulation (DM)** – generates a bitstream based on amplitude changes, which is then line encoded.

4. Optional scrambling (B8ZS or HDB3) is provided for AMI encoding to handle long sequences of zeros.
5. The final waveform is displayed using **Matplotlib** with time on the X-axis and amplitude on the Y-axis.

How to Run the Code

1. Ensure Python 3.x is installed on your system.
2. Install dependencies using:

   ```
   pip install -r requirements.txt
   ```

3. Run the program using:

   ```
   python encodings.py
   ```

4. Follow on-screen prompts to enter input type, data, and encoding scheme.
5. The corresponding waveform will be displayed in a Matplotlib window.

Output Description

The program plots a step graph representing the encoded signal.
X-axis: Time
Y-axis: Amplitude
Title: Selected encoding scheme
The waveform helps visualize logic transitions and timing of different encoding methods.

References

1. Python Documentation – `https://docs.python.org/3/`
2. Matplotlib Documentation – `https://matplotlib.org/stable/`

Conclusion

This project successfully demonstrates the implementation and visualization of major digital and analog encoding schemes. It provides an interactive platform to understand encoding concepts in data communication.