

# Compte-Rendu de TP : Simulateur de Signaux Électriques

## Table des matières

1. Rappel du sujet : .....	2
2. Analyse du sujet : .....	2
a. Les protocoles à implanter : .....	2
3. Choix techniques effectués : .....	2
4. Résultats et tests : .....	3
5. Conclusion : .....	4
a. Difficultés rencontrées : .....	4
b. Limites du programme : .....	4
c. Perspectives d'évolution : .....	4

## 1. Rappel du sujet :

L'objectif de ce TP est de concevoir et réaliser une application Java permettant de visualiser graphiquement le signal électrique correspondant à une suite de bits (0 ou 1) en utilisant différents protocoles de codage en ligne : NRZ, Manchester, Manchester Différentiel et Miller. L'utilisateur doit pouvoir saisir une chaîne binaire et choisir le mode de codage via une interface graphique.

## 2. Analyse du sujet :

L'objectif de ce travail est de simuler la transmission en bande de base, technique utilisée dans les réseaux locaux où les données binaires sont transmises directement sur le câble via un signal numérique. Le programme doit traduire une séquence logique (bits) en une forme d'onde physique (tensions) selon quatre protocoles de codage spécifiques.

### a. Les protocoles à implanter :

Le simulateur doit gérer les règles strictes de transition pour chaque code :

1. **Code NRZ (No Return to Zero) :**
  - a. Codage du bit 1 par un signal de  $n$  volts.
  - b. Codage du bit 0 par un signal de  $-n$  volts
2. **Code Manchester (Biphase) :** Le bit 1 est codé par un passage de la tension  $n$  à  $-n$  et 0 par le passage en sens inverse.
3. **Code Manchester Différentiel :**
  - a. Similaire au Manchester, il impose une transition systématique au milieu du bit.
  - b. La différence réside dans le début du cycle : le bit 0 est codé par une transition supplémentaire au début de l'intervalle, contrairement au bit 1.
4. **Code Miller :**
  - a. Le bit 1 possède une transition au milieu du temps d'horloge, tandis que le bit 0 n'en a pas.
  - b. Pour éviter la perte de synchronisation sur les suites de 0, une transition est ajoutée à la fin du bit si un 0 est suivi d'un autre 0.

## 3. Choix techniques effectués :

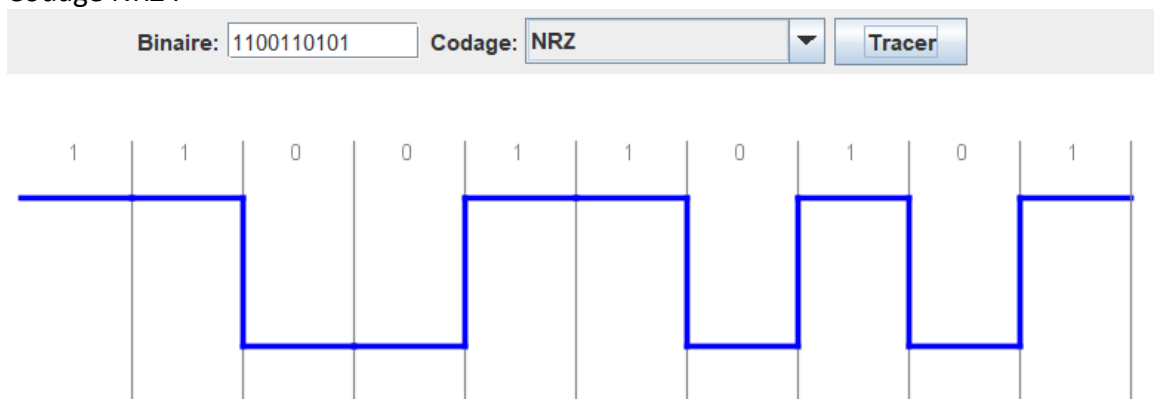
- **Langage :** Java avec les bibliothèques `javax.swing` et `java.awt`.
- **Structure :** Une classe principale `SimulateurSignaux` héritant de `JFrame` contenant une classe interne `DrawingPanel` pour isoler la logique de dessin.
- **Gestion des états :** Utilisation d'une variable `lastY` pour assurer la continuité du signal entre deux bits consécutifs. Cela est crucial pour les codages différentiels ou avec transitions inter-bits.
- **Algorithmes de codage :**
  - **NRZ :** Niveau haut pour '1', niveau bas pour '0'.

- **Manchester** : Transition systématique au milieu du bit (signal montant pour '0', descendant pour '1').
- **Manchester Différentiel** : Transition au milieu du bit, mais la présence ou l'absence de transition au *début* du bit détermine la valeur (0 = transition, 1 = pas de transition).
- **Miller** : Transition au milieu pour un '1'. Pour un '0', la transition n'a lieu qu'entre deux '0' consécutifs.

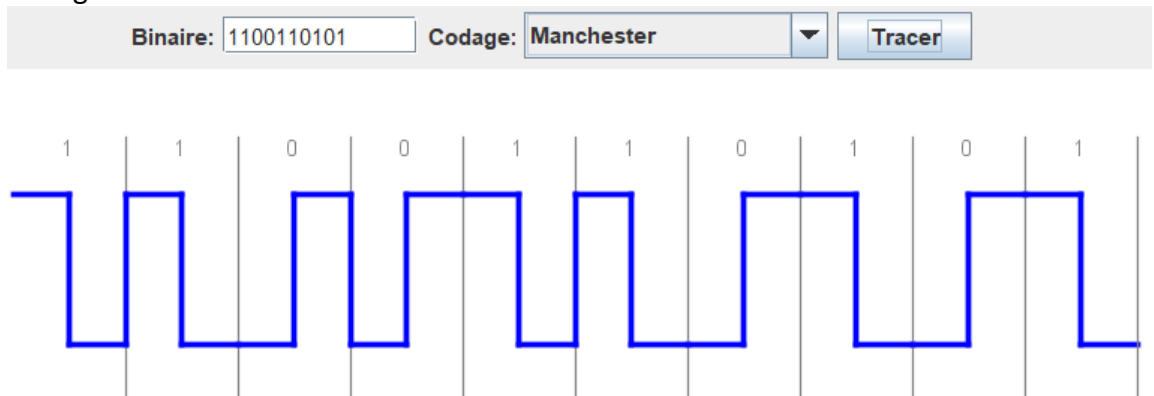
## 4. Résultats et tests :

Le programme a été testé avec la séquence 1100110101 pour chaque mode de codage :

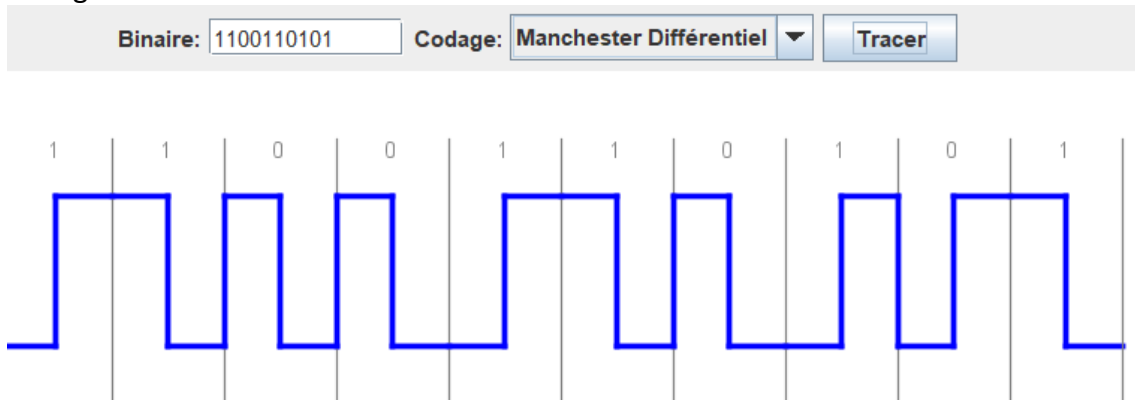
- Codage NRZ :



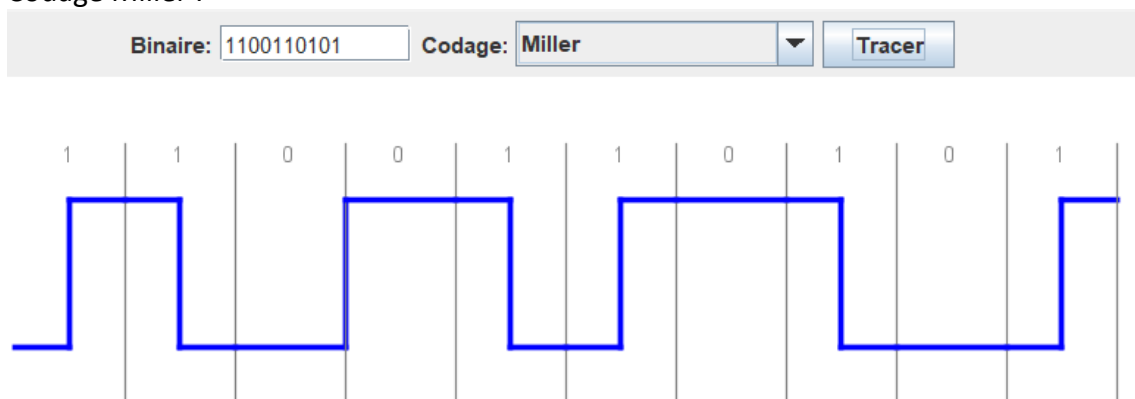
- Codage Manchester :



- Codage Manchester différentiel :



- Codage Miller :



## 5. Conclusion :

### a. Difficultés rencontrées :

La principale difficulté a été la gestion de la continuité du tracé. Il ne suffit pas de dessiner des segments horizontaux ; il faut impérativement tracer des lignes verticales de raccordement si le niveau final du bit précédent est différent du niveau initial du bit actuel. Le codage de Miller a également demandé une attention particulière car il dépend de l'état futur (le bit suivant).

### b. Limites du programme :

- La fenêtre n'est pas "scrollable" : si la chaîne binaire est trop longue, le signal sort de l'écran.
- L'amplitude et la largeur des bits sont fixes.

### c. Perspectives d'évolution :

- Ajout d'une barre de défilement pour les longues séquences.
- Ajout d'autres codages.
- Personnalisation de l'amplitude et de la largeur des bits.