2ème partie

Chapitre 1 :

Introduction à la communication numérique

Pr: M. LAARABI

2022/2023

Plan

1. Introduction
2. Chaîne de communication numérique
3. Codage d’une information numérique
4. Modes de transmissions
5. Nature de liaison

Introduction

Pourquoi ce cours ?

De nos jours, presque toutes les communications sont numériques :

- Téléphones portables : GSM, UMTS (3G), LTE (4G), 5G
- Internet : ADSL
- Télévision : TNT, DVB-S (satellite)
- Radio: RNT (Radio numérique terrestre)

Canaux de transmission :

Câbles coaxiaux, paires torsadées, réseau hertzien, infrarouge, fibres optiques ...

Données numériques

Données représentées sous forme de nombres (souvent binaires 0 ou 1)

- Données numériques
  - Caractères alphanumériques (ex : SMS)
  - Fichier informatique (ex : fichier .doc, .jpeg, .gif, etc ...)

- Données analogiques numérisées
  - Flux audio (ex : voix)
  - Flux vidéo (ex : Skype)

But du cours

- Avoir une compréhension générale des mécanismes permettant l'envoi et la réception de ces données numériques
- Adaptation des chaînes de transmission aux différents canaux utilisés (modulation)
- Etude théorique de chaînes de transmission idéales

Principe de la transmission numérique

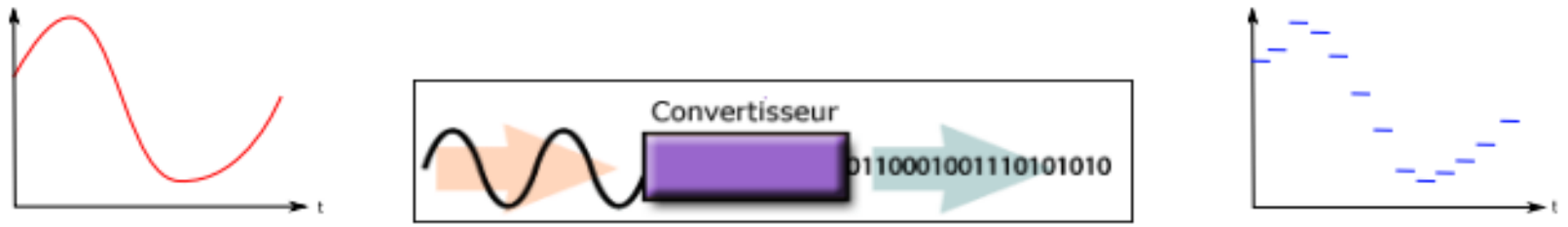
La transmission numérique consiste à faire transiter les informations sur le support physique de communication sous forme de signaux numériques. Ainsi, des données analogiques devront préalablement être numérisées avant d'être transmises.

Un signal analogique est un signal continu qui peu prendre une infinité de valeurs d'amplitude, alors que le signal numérique est un signal discret (discontinu) qui se résume en succession de "0" et de "1". L'objectif de la numérisation est de transformer le signal analogique qui contient une quantité infinie d'amplitudes en un signal numérique contenant lui une quantité finie de valeurs.

La construction d'un signal numérique se décompose en trois étapes fondamentales :

1. Echantillonnage;
2. Quantification;
3. Codage.

La numérisation est faite par un convertisseur analogique-numérique comme l'illustre la figure I.1.



Signal analogique **Figure I .1**- Numérisation Signal numérique

Le nombre d'échantillons composant le signal numérique devra être suffisamment grande pour représenter le signal analogique de départ mais pas trop grand non plus pour ne

pas être trop volumineux. La numérisation est d'autant meilleure que le signal numérique se rapproche du signal analogique initial.

L'étape inverse qui consiste à obtenir un signal analogique à partir d'une suite binaire est essentiellement une opération de filtrage.

I .2 Numérisation

I .2.1 Echantillonnage

L'opération d'échantillonnage consiste à prélever sur le signal analogique des échantillons à intervalles de temps réguliers, c'est la discrétisation de l'axe des abscisses. Il consiste à découper le signal analogique en petites tranches temporelles.

On notera T_e la période avec laquelle on prélève ces échantillons. On note alors $f_e = 1/T_e$ fréquence d'échantillonnage qui correspond au nombre d'échantillons par seconde.

Pour que le spectre du signal échantillonné ne se superpose pas avec le spectre du signal analogique, il faut que la fréquence d'échantillonnage f_e soit au moins deux fois plus grande que la plus grande de la fréquence du signal f_m ce qui donne le théorème de Shannon :

$$f_e > 2f_m \qquad (I.1)$$

Plus la fréquence d'échantillonnage sera grande, plus le nombre d'échantillons sera grand, plus le signal numérique « collera » au signal analogique et donc meilleure sera la numérisation (figure I.2):

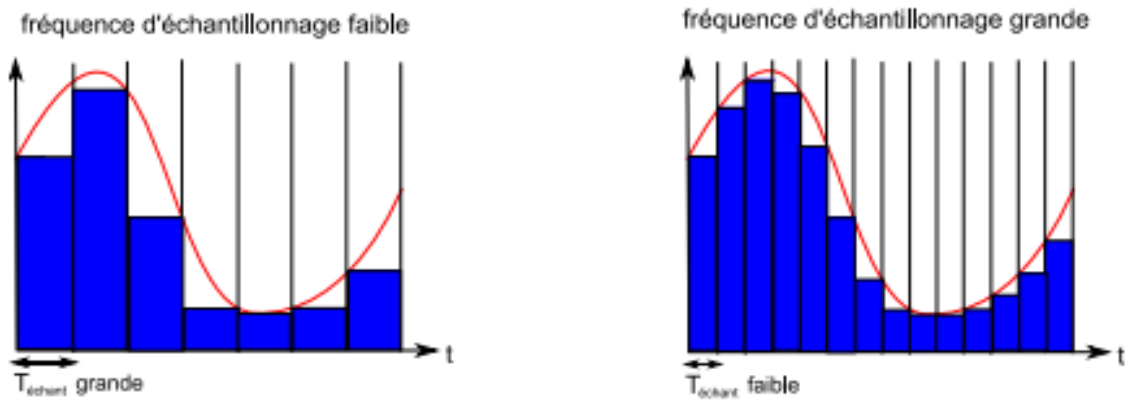


Figure I.2- Echantillonnage d'un signal analogique par deux fréquences

Le signal analogique $s(t)$, continu dans le temps, est alors représenter par un ensemble de valeur discrètes :

$$s_e(t) = s(nT_e) \qquad \text{(I .2)}$$

Avec: n est un entier; T_e est la période d'échantillonnage.

Cette opération est réalisée par un échantillonneur souvent symbolisé par un interrupteur (figure I .3).

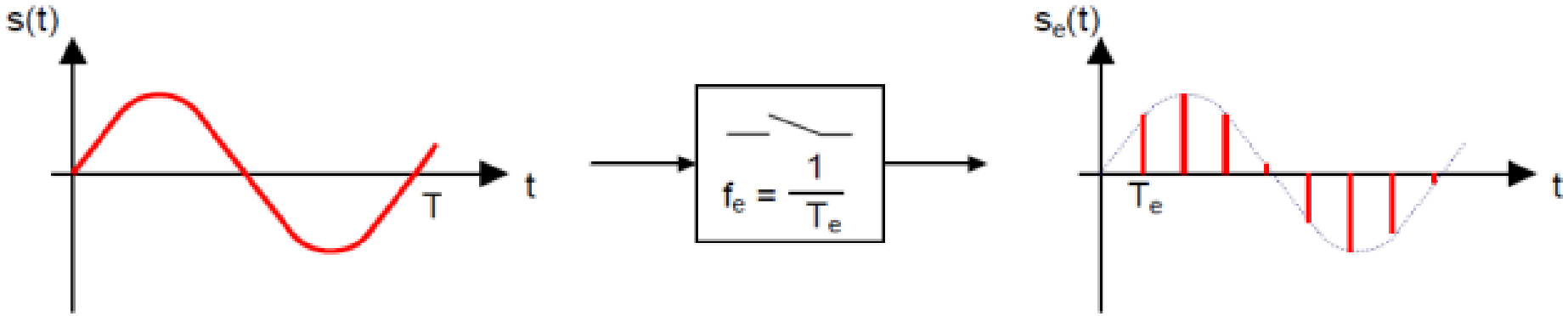


Figure I .3- Echantillonneur

- **Echantillonnage idéal**

L'échantillonnage idéal est modélisé par la multiplication du signal continu $s(t)$ par un peigne de Dirac de période T_e.

$$s_e(t) = s(t)\delta_{T_e}(t)$$
$$= s(t)\delta(t - nT_e) \qquad \text{(I .3)}$$

- **Echantillonnage réel**

En pratique, il est impossible d'obtenir des impulsions de durée quasiment nulle. La modélisation de l'échantillonnage par un peigne de Dirac est donc erronée. En fait, chaque impulsion va avoir une durée très courte τ. L'échantillonnage peut donc être modélisé par la multiplication du signal par une suite de fonction rectangle (ou porte) de largeur τ.

I .2.2 Quantification

Lors de la numérisation, il faut également discrétiser les valeurs de l'amplitude du signal (quantification), il consiste alors à découper l'amplitude du signal en valeurs discrètes. La quantification est alors la discrétisation de l'axe des ordonnées.

Le nombre de valeurs dont on dispose pour définir l'amplitude s'appelle la quantification. Elle s'exprime en « bit ». Plus la quantification est grande, plus l'amplitude du signal numérique sera proche de celle du signal analogique (figure I .4).

Pour une valeur binaire de n bits, l'amplitude crête à crête du signal est découpée en 2^n tranches. Chaque tranche est appelée pas de quantification, donnée par la formule suivante:

$$pas\ de\ quantification = \frac{amplitude\ crête\ à\ crête}{2^n} \qquad (I\ .4)$$

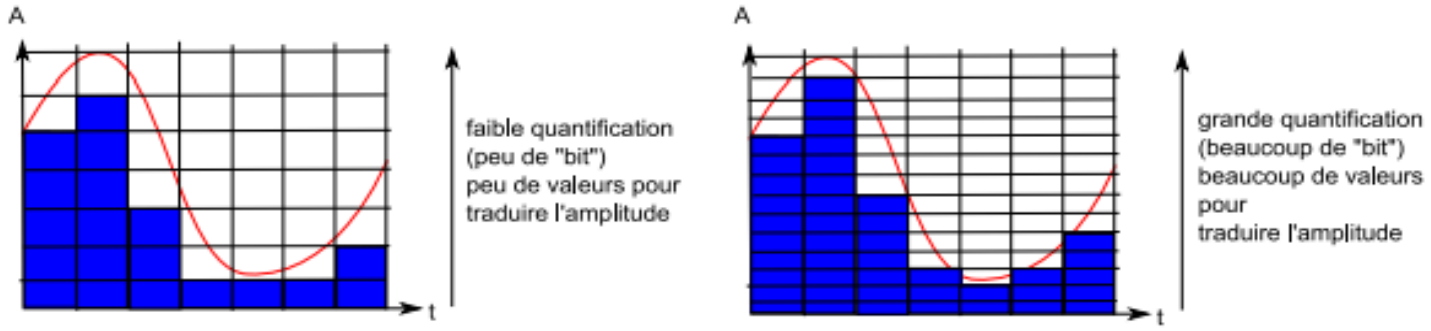


Figure I .4- Quantification

I .2.3 Codage

L'opération de codage consiste à coder les échantillons à l'aide d'un ensemble de combinaisons binaires. Il consiste à associer à chaque tranche découpée une valeur de n bits. Alors, chaque valeur échantillonnée est associée à la valeur binaire de la tranche ou elle se situe ce suivant une certaine loi : arrondi supérieur, arrondi le plus proche, etc…

Voici un exemple de graphe pour une quantification sur 2 bits. L'ensemble des valeurs échantillonnées constitue le signal numérique, comme le représente la figure I .5 :

$code = 10\ 11\ 11\ 11\ 11\ 11\ 11\ 10\ 01\ 00\ 00\ 00\ \ldots.$

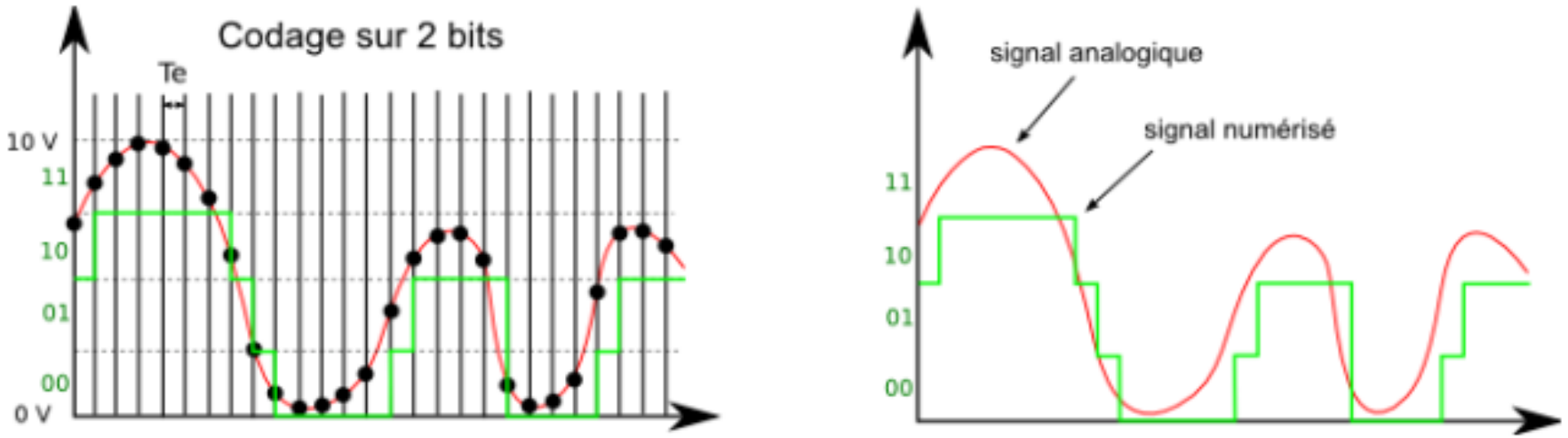


Figure I .5- Codage

Plus la quantification est grande, meilleure sera la numérisation. Regardons la figure I .5, la différence d'amplitude entre le signal numérisé en deux bits et le signal analogique peut être très grande. Regardons la figure I .6, si le même signal est numérisé sur trois bits la différence d'amplitude est très petite, et en quatre bits c'est encore mieux. Alors, plus le nombre de bits de la quantification augmente plus les deux amplitudes sont proches, c'est à dire meilleur est la qualité de la numérisation.

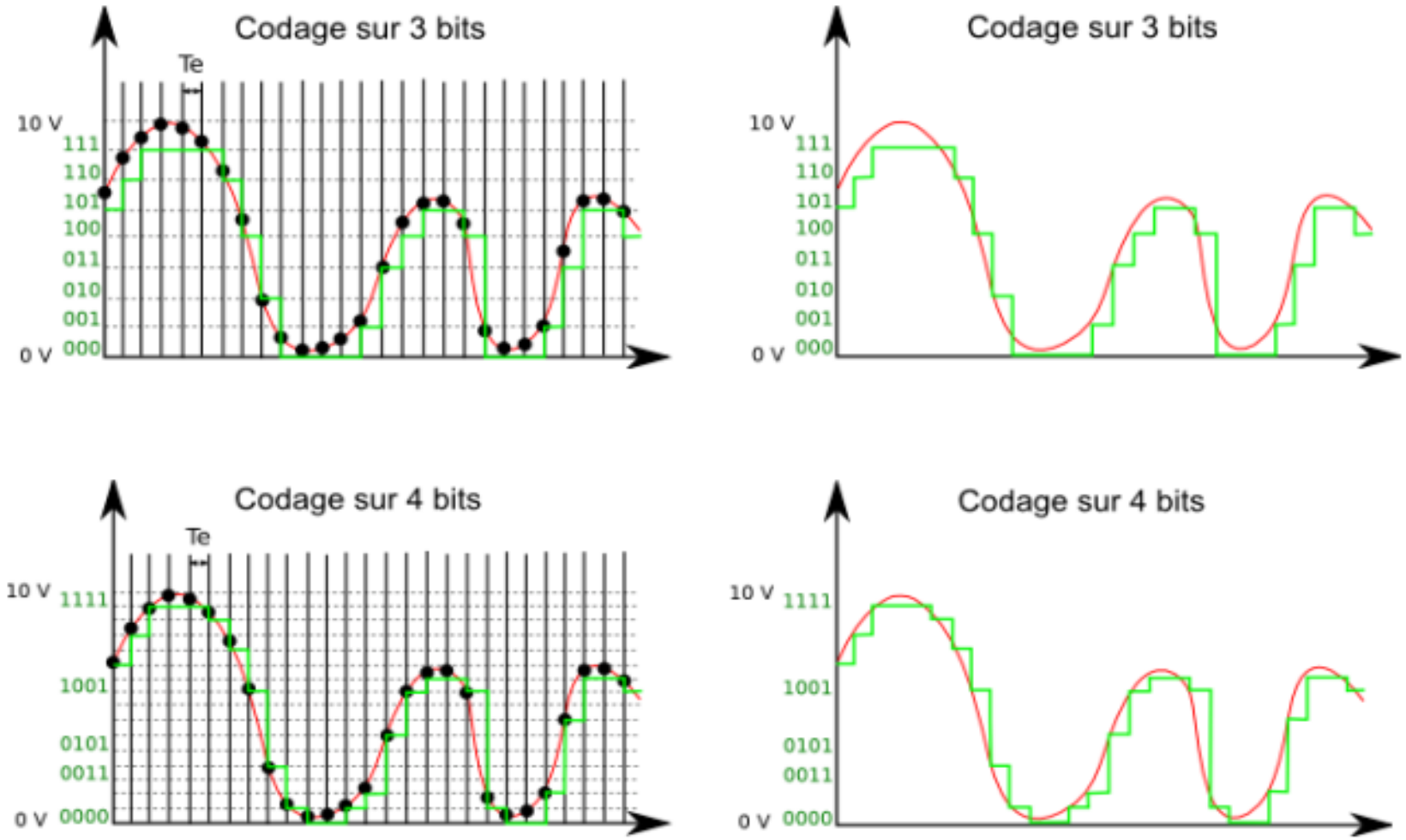


Figure I .6- Exemples de Codage

- **Poids en octets du son**

La limite des valeurs échantillonnées vient du nombre de bits qui vont être nécessaires pour numériser le signal car ce nombre sera écrit sur un support de stockage (disque dur, clé USB, DVD…). La capacité de stockage de ces supports est limitée. De plus, il faut penser qu'il faut du temps pour écrire toutes ces données sur un support (durée qui dépend de beaucoup de paramètres : type de support, version du port USB etc….).

Le nombre N d'octets (ensemble de 8 bits) nécessaires pour «décrire» numériquement une minute de son est:

$$Poids_{bits} = n_e \times quantification \times nombre\ de\ voies \qquad (I.5)$$

$$Poids_{octets} = \frac{Poids_{bits}}{8} \qquad (I.6)$$

Avec

- n_e est le nombre d'échantillons. Pour une durée d'enregistrement Δt et une période d'échantillonnage T_e, le nombre d'échantillons est donné par:

$$n_e = \frac{\Delta t}{T_e} = \Delta t \times f_e \qquad \text{(I .7)}$$

- $quantification$ est le nombre de bits

$$\text{- } nombre\ de\ voies = \begin{cases} 2 & si\ le\ son\ est\ stéréo \\ 1 & si\ le\ son\ est\ mono \end{cases} \qquad \text{(I .8)}$$

Exemple

Dans le cas d'un CD audio, la numérisation se fait sur 2×16 bits (stéréo) avec une fréquence d'échantillonnage de 44.1 kHz, la place théorique occupée sur un CD par une minute de musique non compressée est:

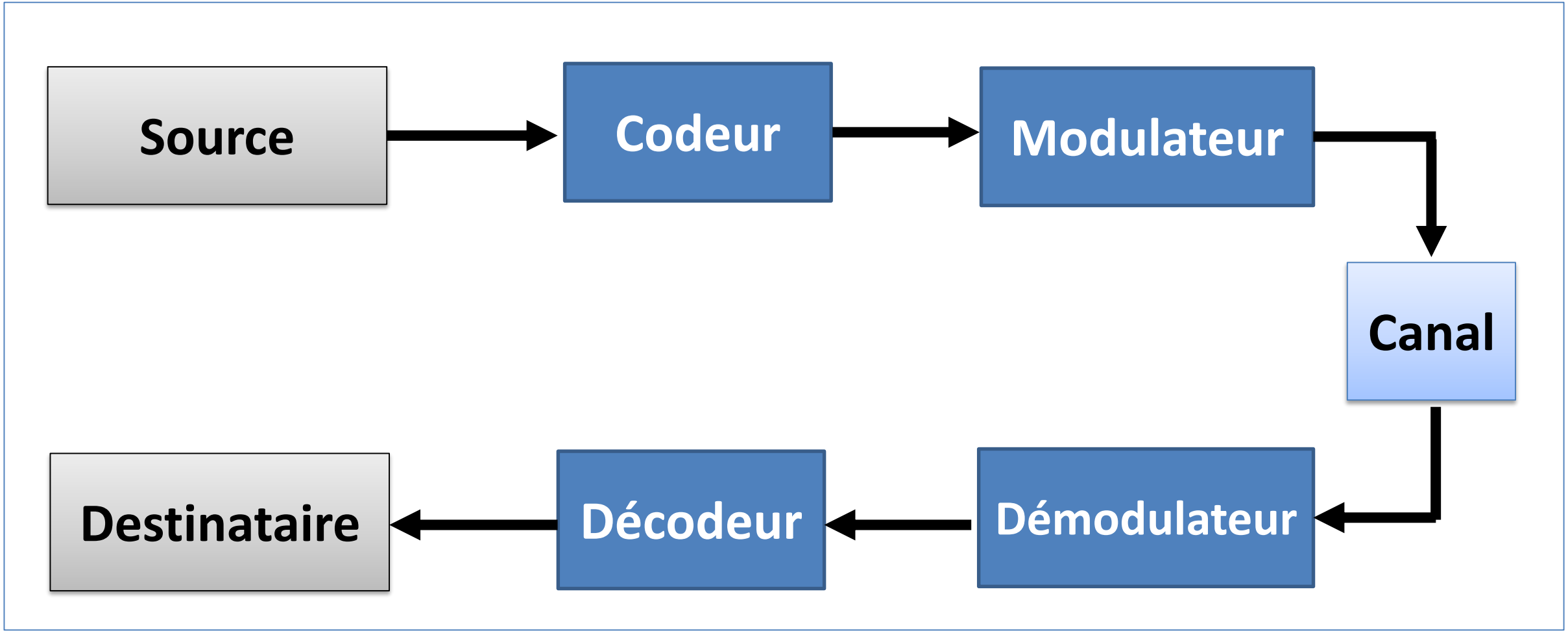
$Poids_{bits} = 60 \times 441\,000 \times 16 \times 2 = 846\,720\,000$ bits.

Plan

1. Introduction
2. **Chaîne de communication numérique**
3. Codage d’une information numérique
4. Modes de transmissions
5. Nature de liaison

2. Chaîne de communication numérique

Schéma général



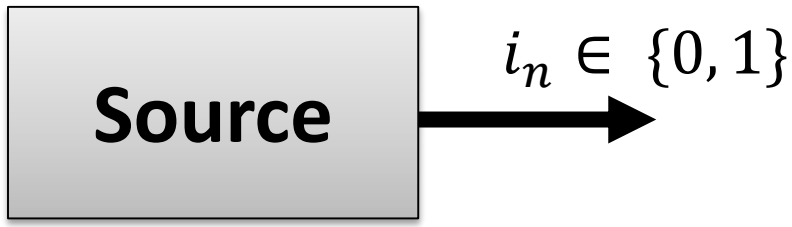
La source:

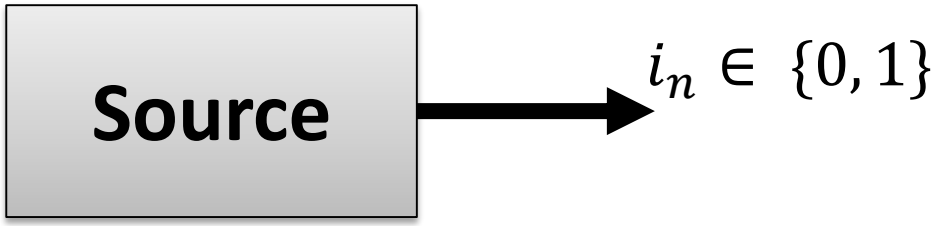
On considère une source numérique à transmettre:

- Suite de symboles prenant un nombre fini de valeurs (e.g. deux valeurs pour une source binaire 0 et 1),
- Cadencée par une horloge de période Ts.

◭ Données discrètes : texte, numéros ...

◭ Données analogiques numérisées : image, voix, vidéo ...



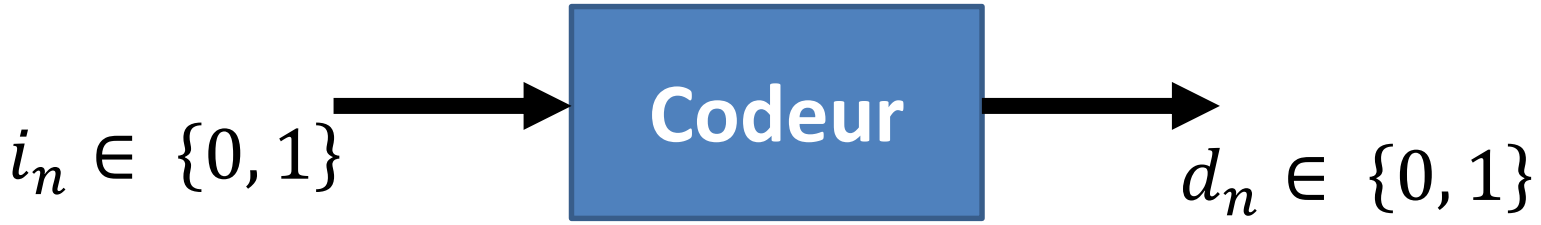


- **Codage source** : représente le message sous la forme la plus économique possible en terme de nombre de bits (symboles codés par des mots de longueurs variables, algorithme de Huffman…).
-

Le codeur:

Transformer le signal numérique brut en un nouveau signal numérique optimisé et robuste aux erreurs.

- **Codage canal:** ajoute des informations permettant au récepteur de reconstituer le message malgré les erreurs éventuellement apparues à cause du bruit sur le canal.

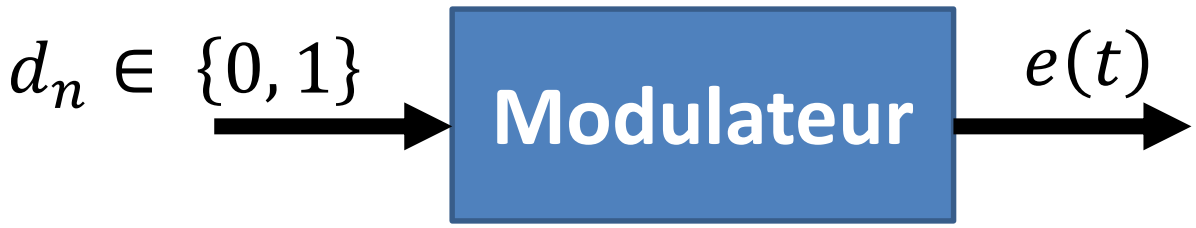


Le modulateur :

Transformer le signal numérique en un signal physique (onde électromagnétique, signal électrique, etc...) qui puisse être transmis sur le canal de transmission.

◭ Transmettre le maximum de données avec une fiabilité maximale

◭ S'adapter au canal de transmission utilisé.

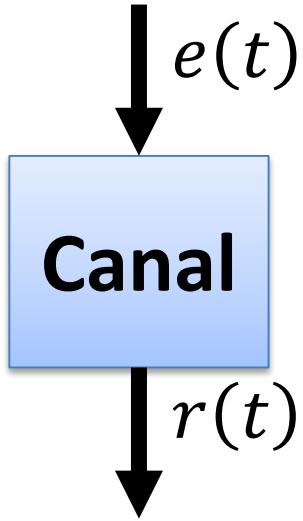


La modulation a pour rôle d'adapter le spectre du signal au canal (milieu physique) sur lequel il sera émis.

Le canal de transmission:

Le support de la transmission est un canal physique:

- Câbles (paire torsadée, câble coaxial. . .),
- Transmission électromagnétique en espace libre (canal hertzien – TNT, Wifi …),
- Optique en espace libre (infrarouge, …),
- Fibre optique . . .



⚠ Propriétés physiques différentes selon le canal utilisé : bande passante, débit maximal, etc...

⚠ Eventuellement source d'erreurs (bruit, perte de données, etc...)

- Le canal est supposé linéaire, invariant dans le temps (filtre linéaire),
- Il apporte des perturbations au signal (bruit additif).

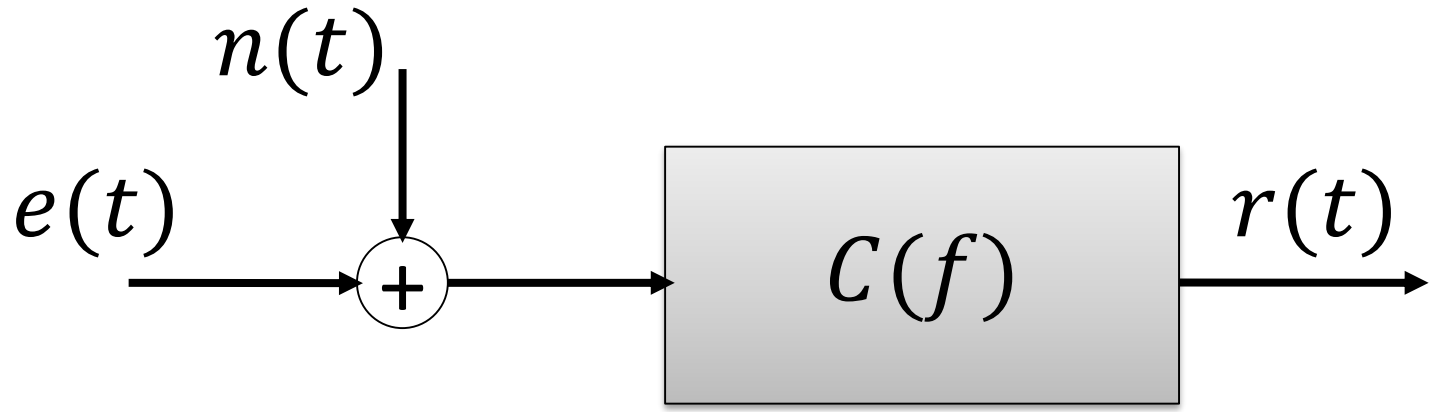


Schéma de modélisation du canal physique

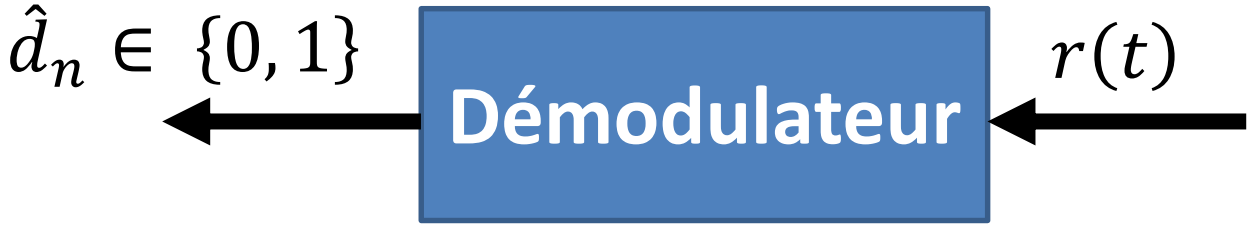
- $e(t)$ le signal émis, $n(t)$ le bruit additif (aléatoire), $C(f)$ la fonction de transfert du canal,
- Le signal reçu : $r(t) = [\boldsymbol{e(t)} + \boldsymbol{n(t)}] * c(t)$, où $'' * ''$ représente la convolution.

Rappel: $\boldsymbol{x(t) * y(t) = \int_{-\infty}^{+\infty} x(t-\tau)\, y(\tau)\, d\tau}$

Le démodulateur:

Transformer le signal physique reçu pour retrouver le signal numérique envoyé.

- Echantillonnage, détection, élimination du bruit.
- Parfois difficile s'il y a eu trop de bruit ou de perte de données.

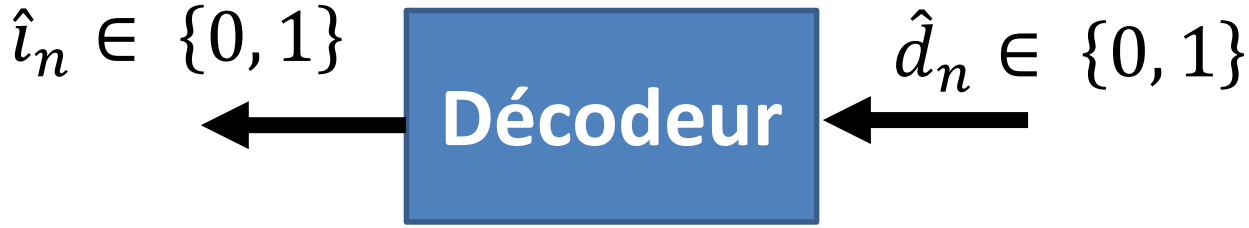


La démodulation est la même que la fonction modulation, mais d'une manière inverse.

Le décodeur:

À partir de l'estimation du signal numérique envoyé, reconstruction du message original (texte, son, image, vidéo ...)

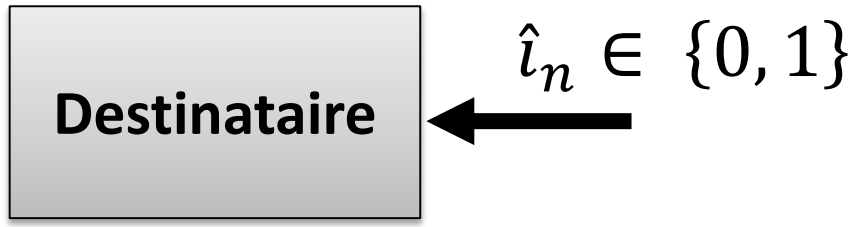
◮ Inversion des étapes de codage source et codage canal



Le destinataire:

Recevoir l'information envoyée par le décodeur sous forme des bits en binaires (0 et 1),

- Données discrètes : texte, numéros ...
- Données analogiques numérisées : image, voix, vidéo ...



Plan

1. Introduction
2. Chaîne de communication numérique
3. **Codage d'une information numérique**
4. Modes de transmissions
5. Nature de liaison

3. Codage d'une information numérique

L'information que l'on désire transmettre doit être adaptée au mode de fonctionnement des éléments utilisés (ordinateur, automate, console de jeu…). Il faut donc la coder.

Exemple de l'électricité.

Nous connaissons les deux états d'une lampe électrique : La lampe est **allumée** ou **éteinte**.

Le langage machine utilise deux symboles : le 1 et le 0.

- "1" signifie passage du courant
- "0" signifie absence de courant

Il s'agit du système de numération binaire, chaque bit prend les valeurs 0 ou 1. Grâce au système binaire, on peut donc manipuler des nombres entiers, voire réels. Il existe bien sûr d'autres systèmes de numération.

Comment fera-t-on pour envoyer : quoi?

Codage de source :

1. On remplace chaque caractère par un nombre.

Exemple : Si on envoie le code 00110111101110011111.

Il faut définir un repère. Par exemple, le nombre de bits de chaque caractère est 4. En calculant 0011 0111 1011 1001 1111 on obtient 3, 7, 11, 9 et 15.

2. Pour savoir ce que cela veut dire, il faut définir un système de codage.

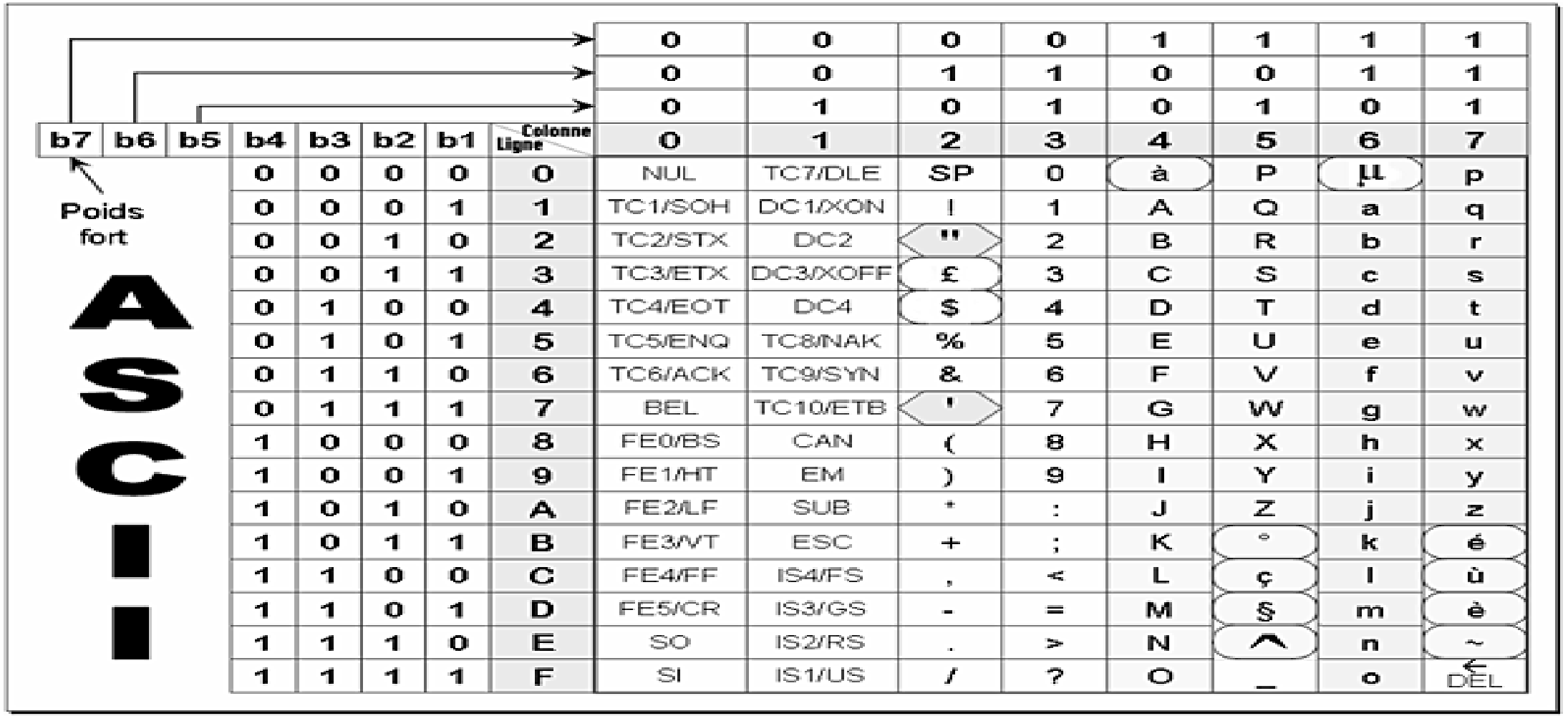
En informatique, des codes ont été normalisés afin que tous les systèmes puissent se comprendre.

Par exemple:

- ASCII (American Standard Code for Information Interchange).
- Unicode
- …

Le code ASCII :

Créé en 1960, il représente les caractères sur 7 bits (c'est-à-dire 128 caractères possibles, de 0 à 127).



b7	b6	b5	b4	b3	b2	b1	Colonne / Ligne	0	0	0	0	1	1	1	1
								0	0	1	1	0	0	1	1
								0	1	0	1	0	1	0	1
								0	1	2	3	4	5	6	7
			0	0	0	0	0	NUL	TC7/DLE	SP	0	à	P	µ	p
			0	0	0	1	1	TC1/SOH	DC1/XON	!	1	A	Q	a	q
			0	0	1	0	2	TC2/STX	DC2	"	2	B	R	b	r
			0	0	1	1	3	TC3/ETX	DC3/XOFF	£	3	C	S	c	s
			0	1	0	0	4	TC4/EOT	DC4	$	4	D	T	d	t
			0	1	0	1	5	TC5/ENQ	TC8/NAK	%	5	E	U	e	u
			0	1	1	0	6	TC6/ACK	TC9/SYN	&	6	F	V	f	v
			0	1	1	1	7	BEL	TC10/ETB	'	7	G	W	g	w
			1	0	0	0	8	FE0/BS	CAN	(	8	H	X	h	x
			1	0	0	1	9	FE1/HT	EM	)	9	I	Y	i	y
			1	0	1	0	A	FE2/LF	SUB	*	:	J	Z	j	z
			1	0	1	1	B	FE3/VT	ESC	+	;	K	°	k	é
			1	1	0	0	C	FE4/FF	IS4/FS	,	<	L	ç	l	ù
			1	1	0	1	D	FE5/CR	IS3/GS	-	=	M	§	m	è
			1	1	1	0	E	SO	IS2/RS	.	>	N	^	n	~
			1	1	1	1	F	SI	IS1/US	/	?	O	_	o	DEL

Poids fort

ASCII

Le code Unicode :

- Système de codage des caractères sur 16 bits mis au point en 1991.
- Il permet de représenter n'importe quel caractère par un code sur 16 bits, indépendamment de tout système d'exploitation ou langage de programmation.
- Il regroupe ainsi la quasi-totalité des alphabets existants (arabe, arménien, grec, latin, ...).
- Il est compatible avec le code ASCII.

Après codage, les informations sont donc toutes représentées sous forme de 0 ou de 1 (ou élément binaire ou bit). C'est ce qu'on appelle la **numérisation.**

On représente cette information par une suite de créneaux.

Une série de 8 bits est appelé **OCTET**

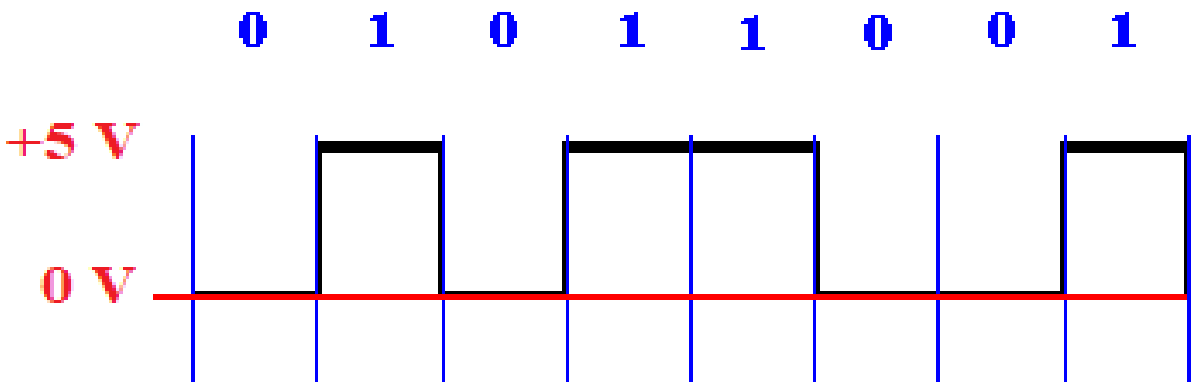
Codage de canal :

Il existe plusieurs types de codage du canal. Parmi ces types on trouve:

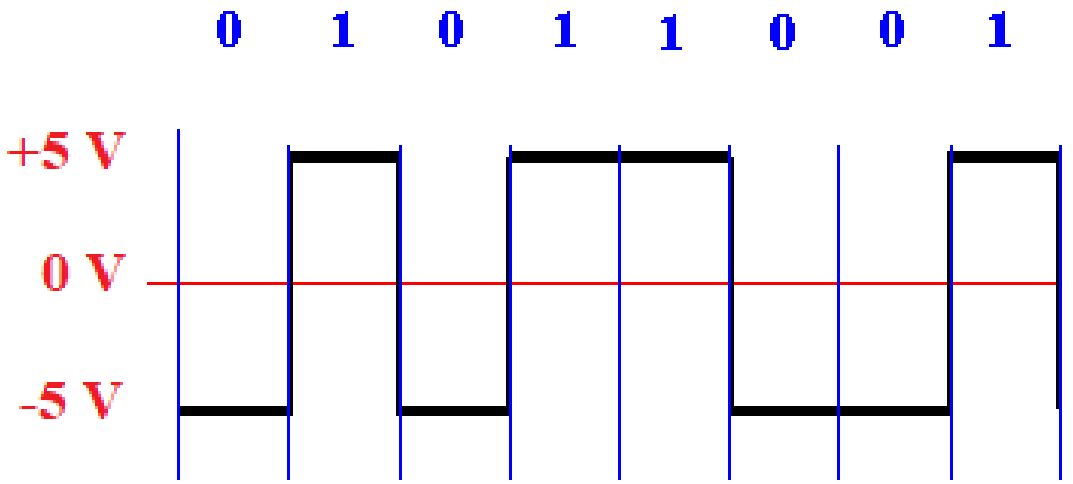
- **NRZ (Non Retour à Zéro)**
- **RZ (Retour à Zéro)**
- **Manchester (ou Biphasé)**
- **Manchester différentiel**
- **AMI**
- **…**

- **NRZ (Non Retour à Zéro)**

Ce type possède deux cas possible: Unipolaire et Bipolaire



NRZ Unipolaire



NRZ Bipolaire

Les tensions $+5\ V$ et $-5\ V$ sont juste des exemples, on peux avoir d'autres valeurs.

Le codage NRZ est souvent utilisé entre l'ordinateur et ses périphériques.

Le codage Non-Retour à Zéro (NRZ)

Le codage NRZ est un codage à deux états, un niveau logique '1' est représenté par une tension +V, et un niveau logique '0' par une tension –V, pendant la durée T d'un bit. Ce codage est illustré dans la figure 13 (a) et résumé par l'équation 11.

Ce codage a l'avantage d'être très facile à mettre en œuvre.

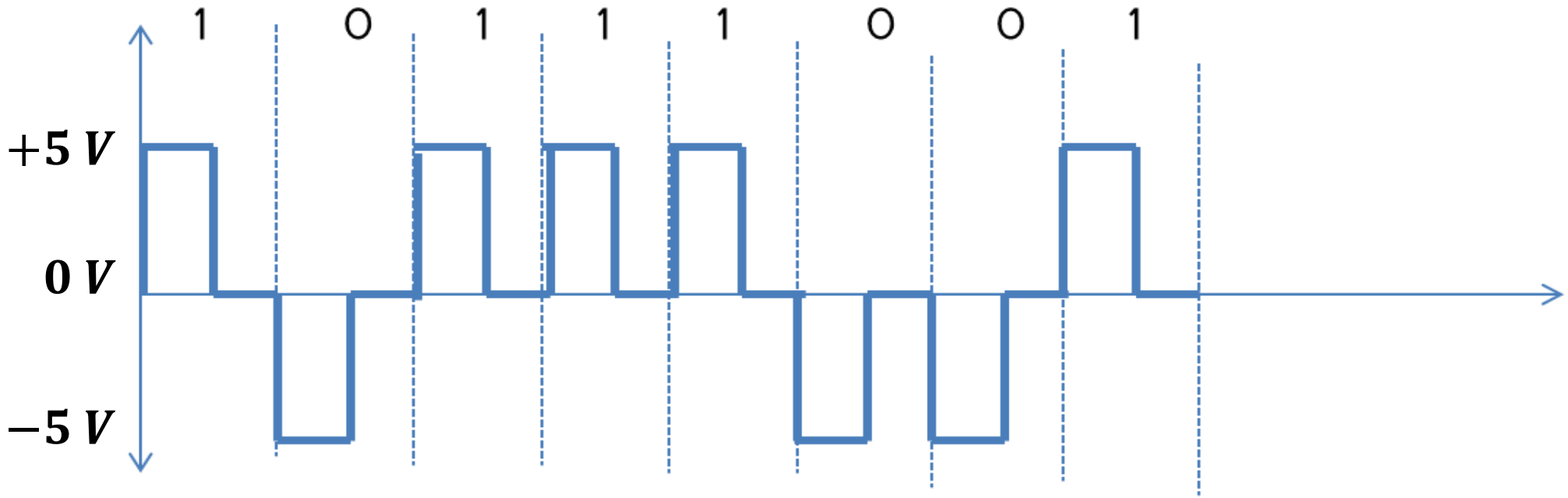
$$\begin{cases} +V & si\ bit = '1' \\ -V & si\ bit = '0' \end{cases} \qquad \text{(Eq 11)}$$

- **RZ (Retour à Zéro)**

La règle est :

Le signal retourne à la valeur zéro après chaque pulse, même s'il y a une succession de deux zéros ou de uns binaires.

Exemple pour la suite binaire: 10111001



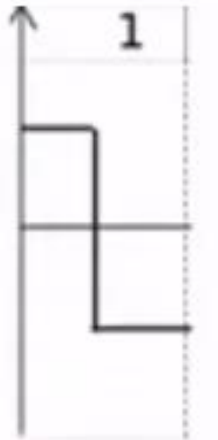
Manchester

On observera qu'une suite de "0" ou de "1" ne produisent pas un signal constant.

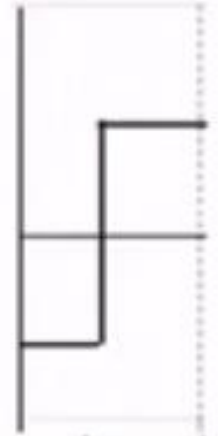
Codage Manchester -> transition au milieu du bit.

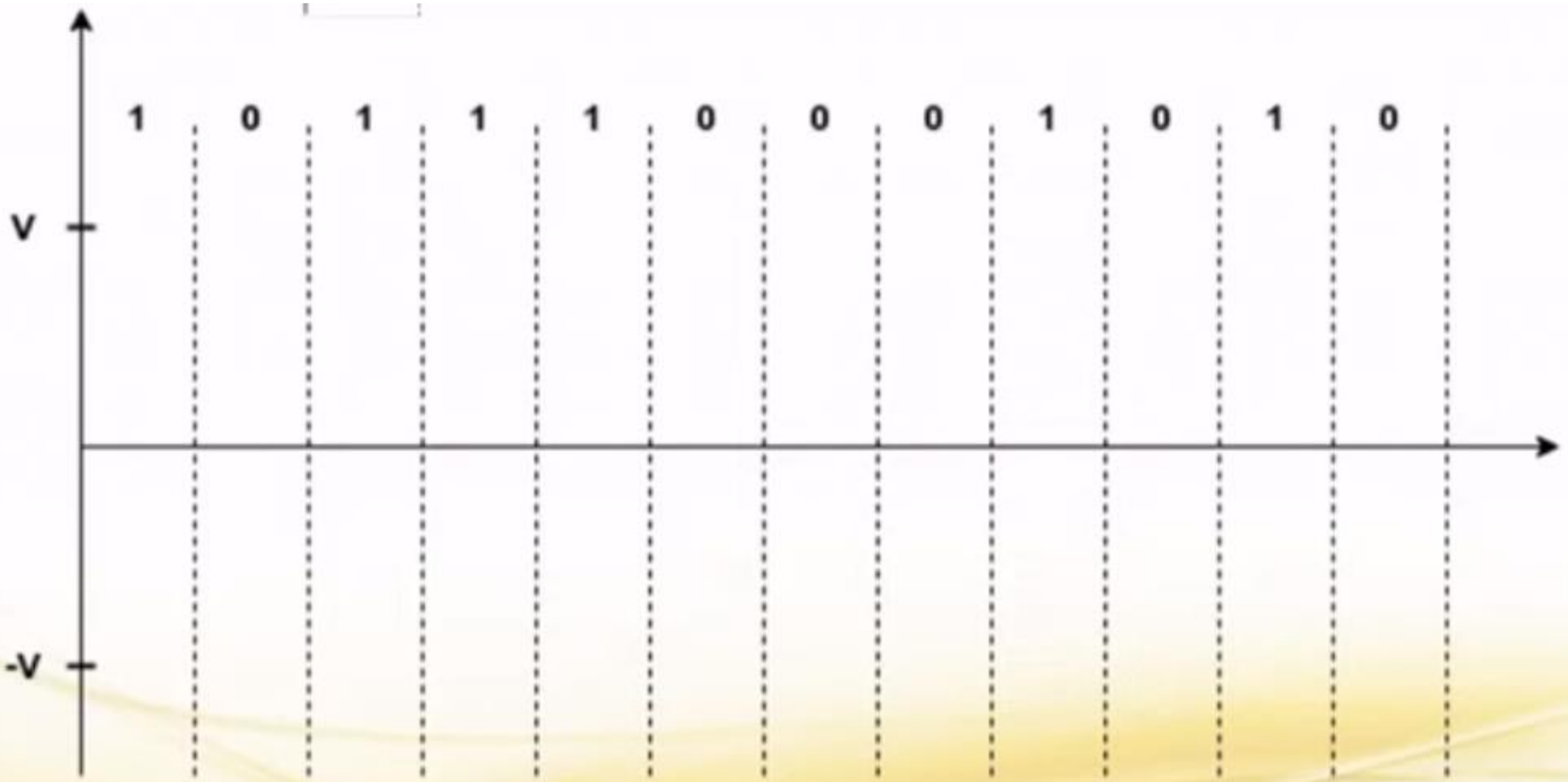
Cela se traduit par :

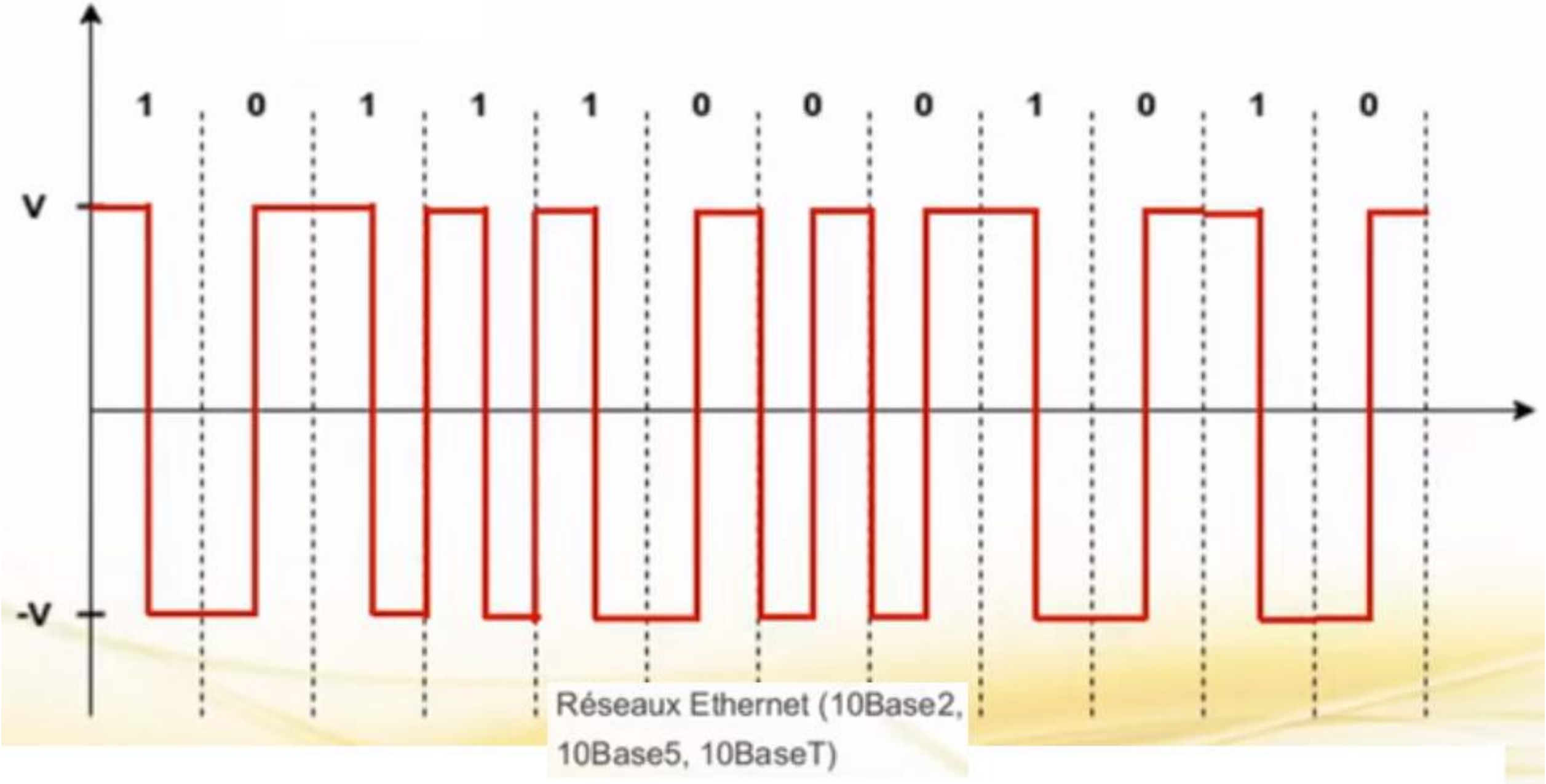
- bit 1 -> transition de haut en bas



- bit 0 -> transition de bas en haut




1 0 1 1 1 0 0 0 1 0 1 0
V
-V



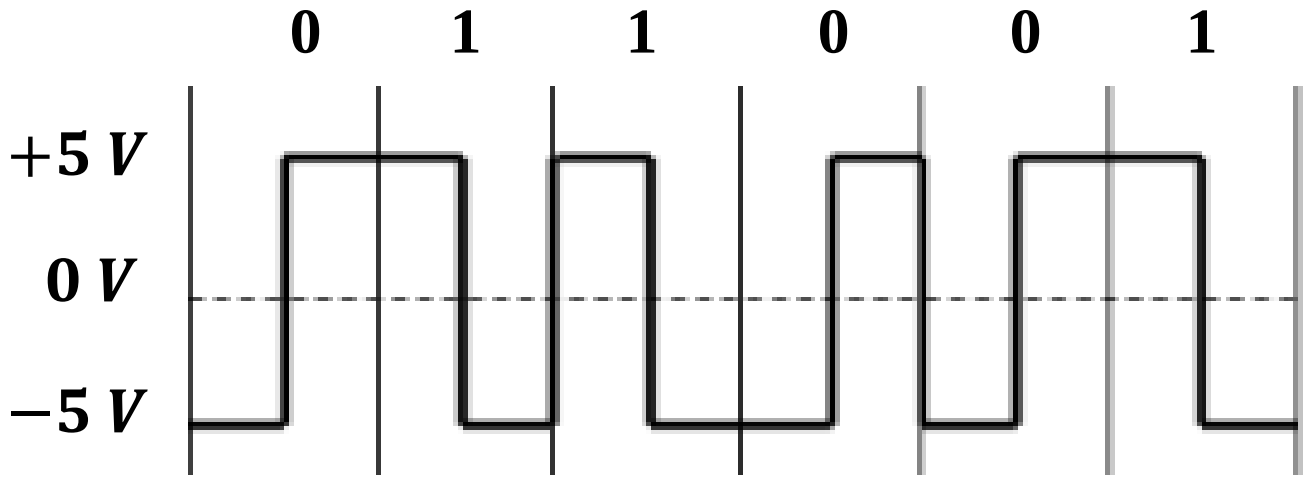
Réseaux Ethernet (10Base2, 10Base5, 10BaseT)

- **Manchester (ou Biphasé)**

La règle est :

- **0** logique: transition du niveau bas vers le niveau haut
- **1** logique: transition du niveau haut vers le niveau bas

Exemple pour la suite binaire: 011001



Ce codage est celui adopté pour les réseaux Ethernet.

Codage Manchester (biphasé)

Ce codage est basé sur le principe de transitions systématiques au milieu de chaque durée T d'un bit (équation 14, figure 14 (a)). Il consiste à représenter le bit '0' par une transition négative qui descend du niveau haut vers le niveau bas en demi-période T. Tant dis que le bit '1' est représenté par une transition positive qui remonte du niveau bas vers le niveau haut en demi-période T.

$$\begin{cases} Transtion\ montante \\ Transition\ descendante \end{cases} \quad \begin{matrix} si\ bit = '1' \\ si\ bit = '0' \end{matrix} \qquad \text{(Eq 14)}$$

❖ **Exemple :** Soit le signal suivant : 1001000

Codage Manchester ***?***

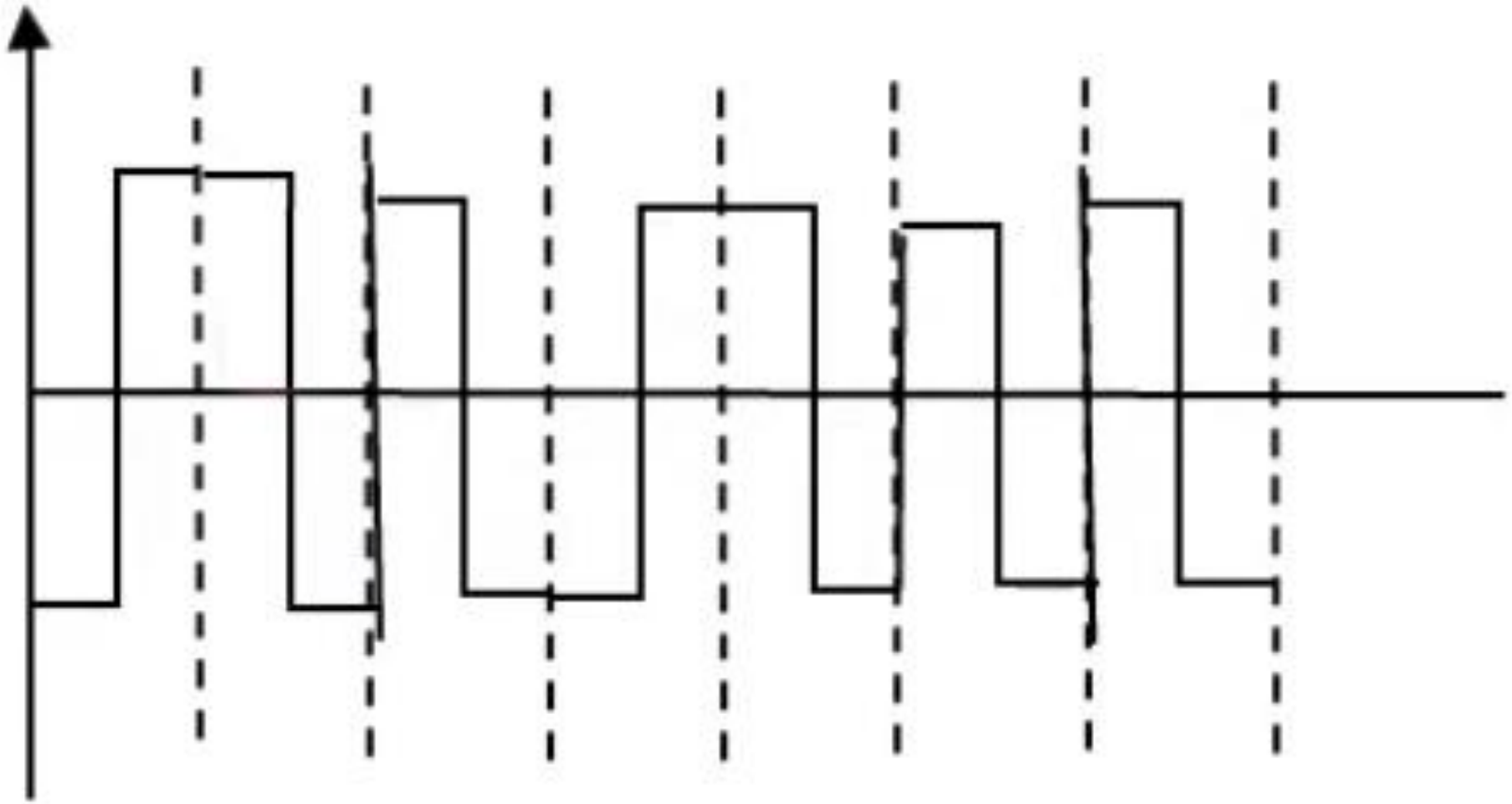
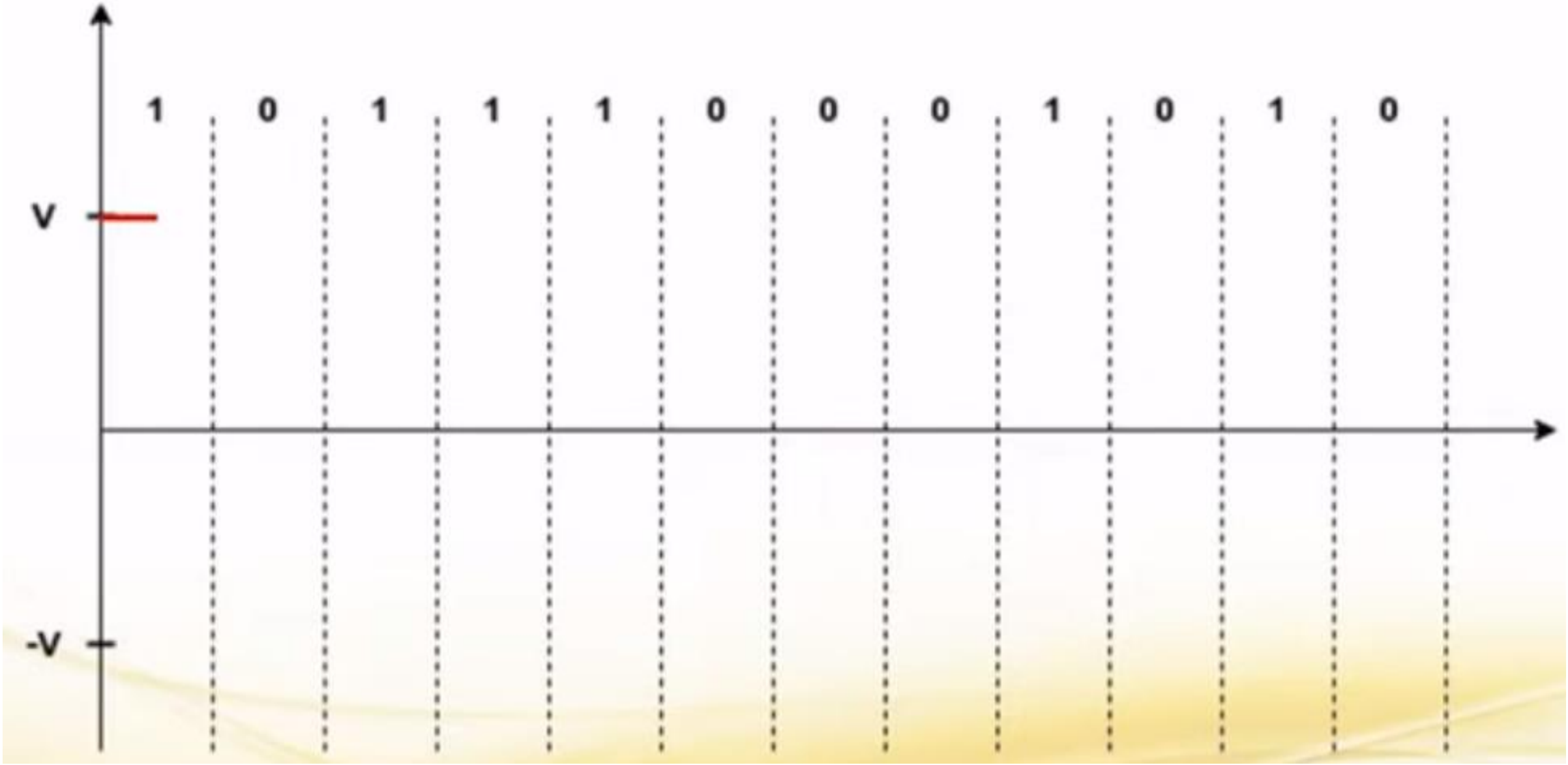
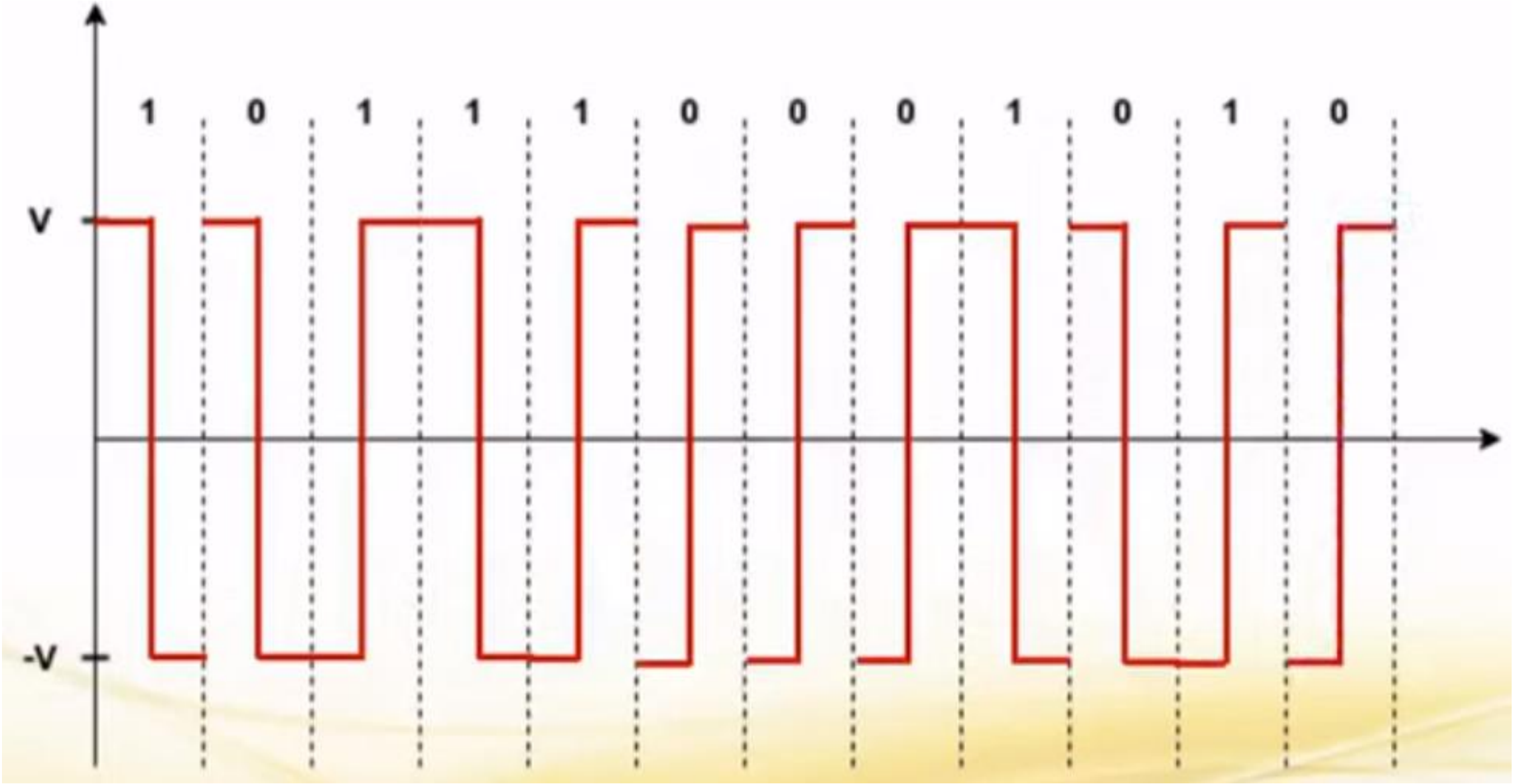


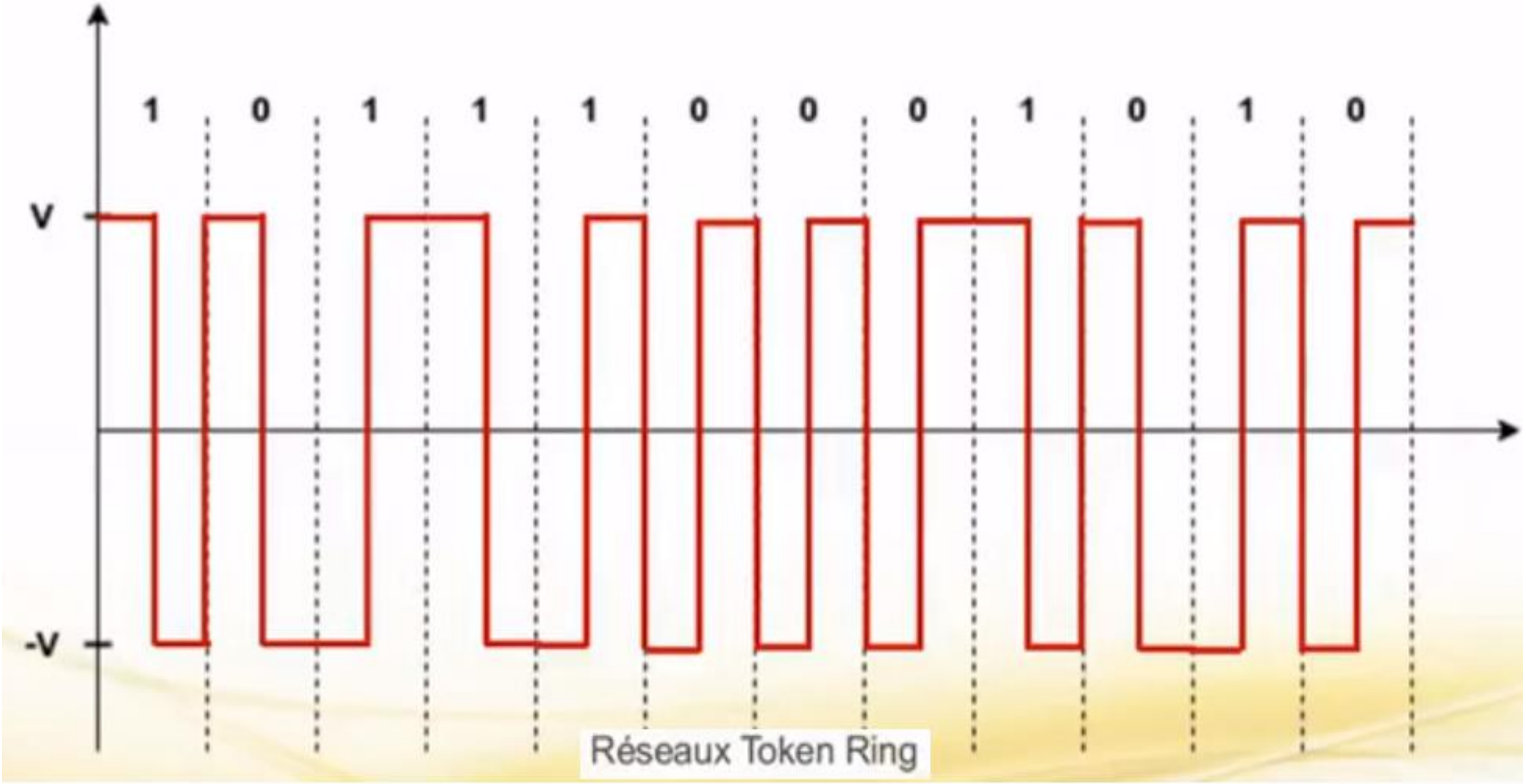
Figure 14 (a). Codage Manchester

Manchester Différentiel

- bit 1 : inverse le sens de la transition (pas de changement du début de voltage)
- bit 0 : la transition est de même sens que la précédente (changement du début de voltage)


1 0 1 1 1 0 0 0 1 0 1 0
V
-V


1 0 1 1 1 0 0 0 1 0 1 0
V
-V


1 0 1 1 1 0 0 0 1 0 1 0
V
-V
Réseaux Token Ring

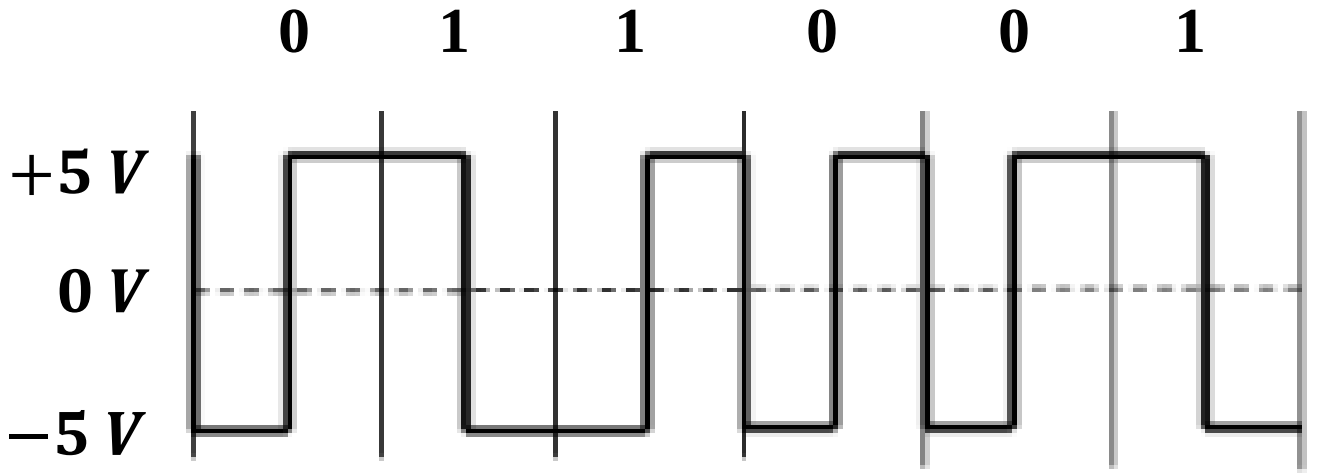
- **Manchester différentiel**

La règle est :

- **0** logique: Transition dans le même sens que la précédente au début de l'intervalle
- **1** logique: Transition dans le sens inverse de la précédente au milieu de l'intervalle

Exemple pour la suite binaire: 011001

Manchester différentiel



Codage Manchester Differentiel

Dans ce codage chaque transition est codée par rapport à la transition précédente (équation 15, figure 14 (b)). Si le bit à coder est '0' alors la transition est de même sens que la transition précédente, sinon si le bit à coder est '1' alors la transition est dans le sens opposée que la transition précédente.

$$\begin{cases} Inverser\ le\ sens\ de\ transition & si\ bit = '1' \\ Maintenir\ le\ sens\ de\ transition & si\ bit = '0' \end{cases} \qquad \text{(Eq 15)}$$

❖ **Exemple :** Soit le signal suivant : 1001011

Codage Manchester Differentiel ***?***

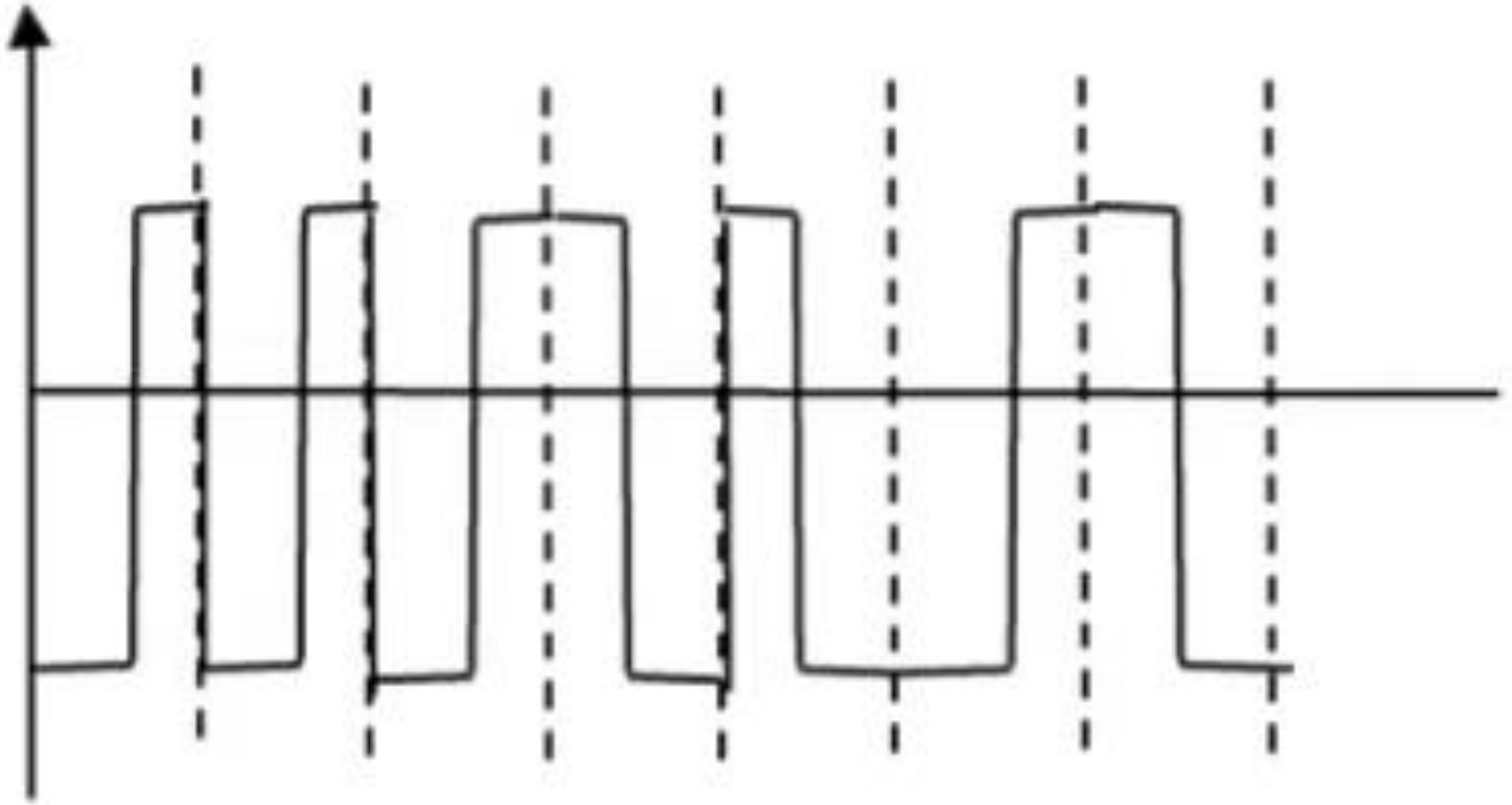


Figure 14 (b). Codage Manchester Differentiel

Codage Miller

Ce codage permet de réduire la bande passante nécessaire pour le codage Miller (Figure 14 (c)), il est construit de la façon suivante :

- Si le bit à coder est '0' alors absence de transition.
- Si le bit à coder est '1' alors la transition est au milieu de la période du bit T.
- Si le bit à coder est '0' précéder d'un autre '0', alors transition à la fin du premier bit.

Exemple : Soit le signal suivant : 1001011

Codage Miller ***?***

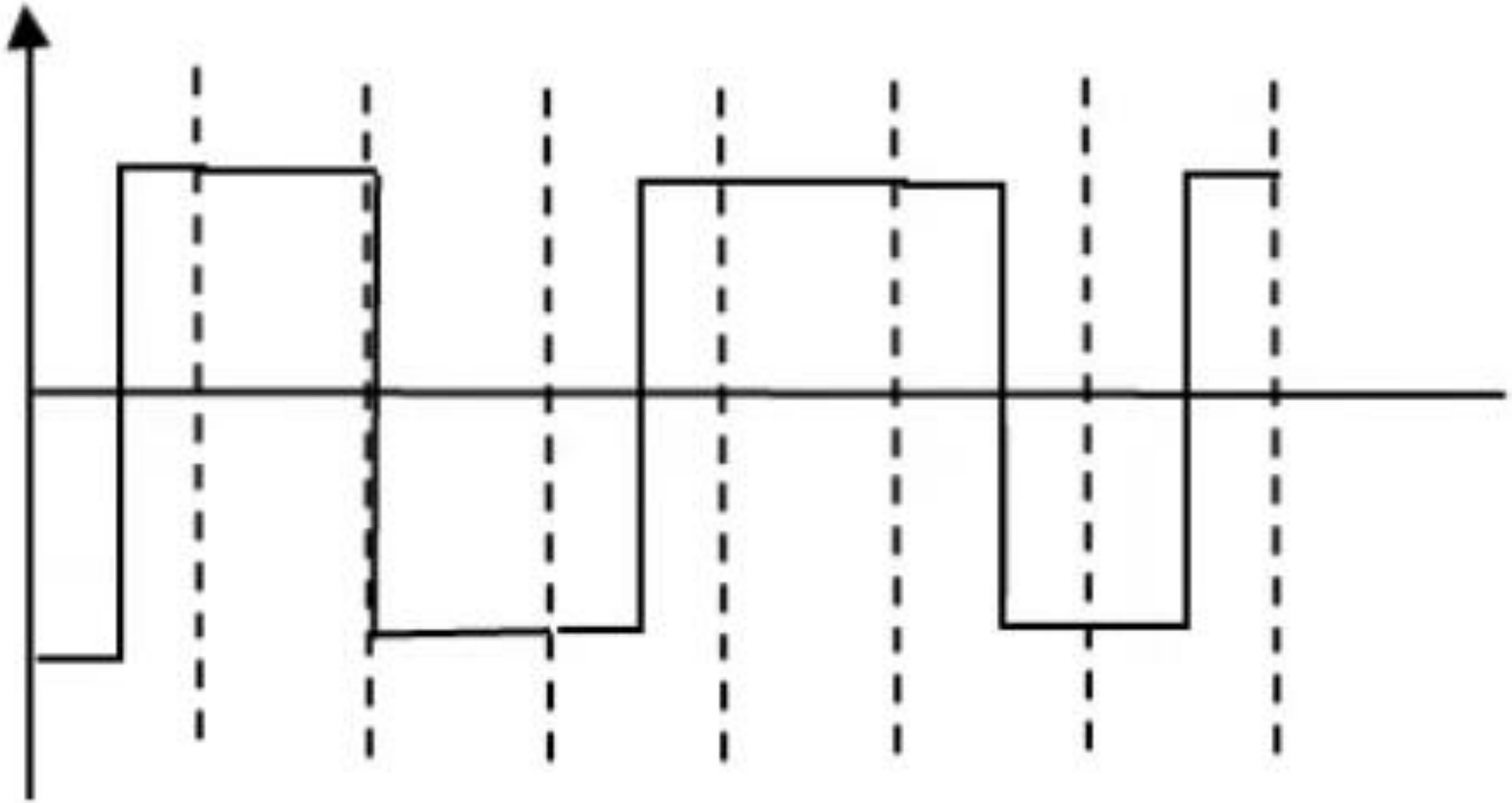


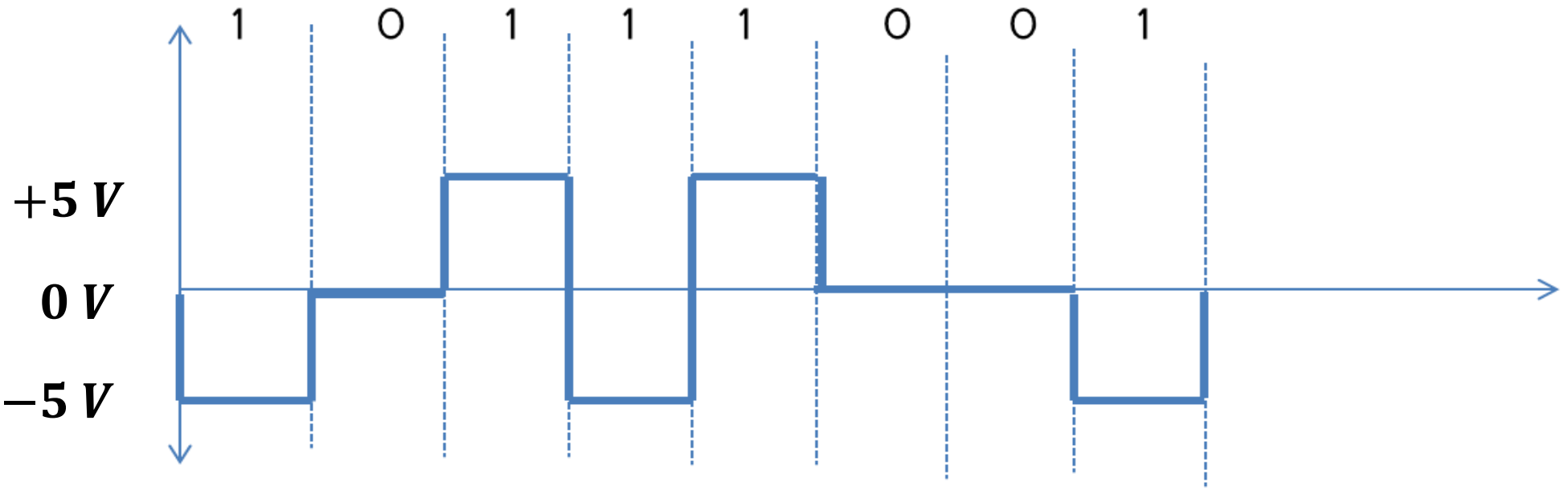
Figure 14 (c). Codage Miller

▪ AMI (Alternate Mark Inversion)

La règle est:

- ✓ Le niveau logique "0" est traduit par un signal nul en ligne : 0 V
- ✓ Le niveau logique "1" est traduit, soit par +5V, soit par -5V.
- ✓ Avec cette contrainte que les "+5" et les "-5" doivent **alterner**.

Exemple pour la suite binaire: 10111001



Exemple d'application 1:

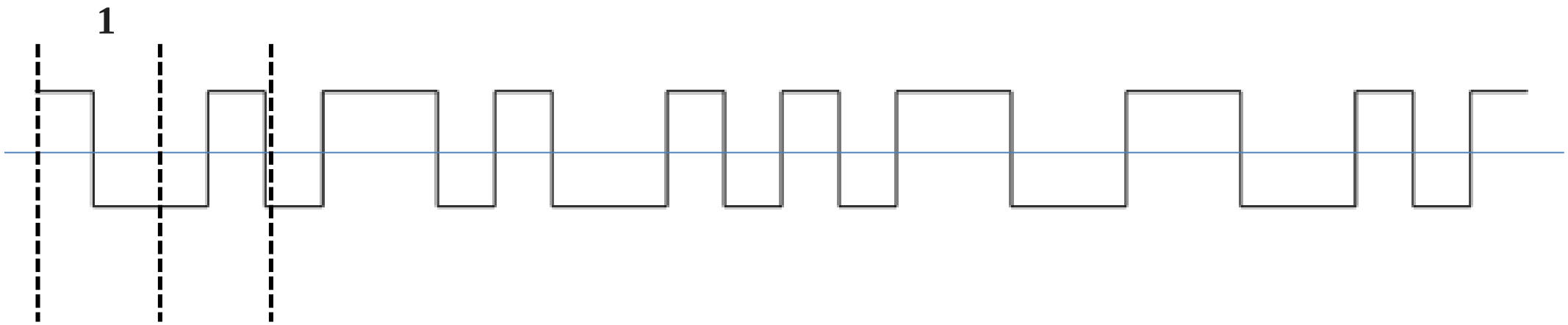
Représenter la suite binaire **01001110001** selon les codes suivants:

- **NRZ (Non Retour à Zéro)**
- **RZ (Retour à Zéro)**
- **Manchester (ou Biphasé)**
- **AMI**

Rq: Les niveaux des tensions considérés sont: $\mathbf{0\ V, -5\ V}$ **et** $\mathbf{+5\ V}$**.**

Exemple d'application 2:

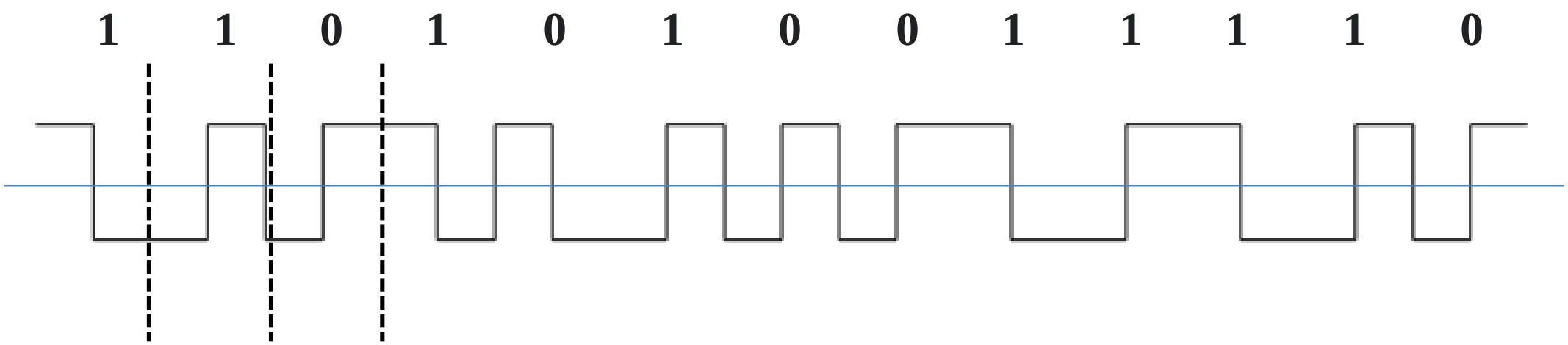
Extraire la suite binaire du signal suivant sachant qu'il est codé avec Manchester différentiel :



Rappel : Règle de Codage Manchester différentiel

- **0** logique: Transition dans le même sens que la précédente au début de l'intervalle
- **1** logique: Transition dans le sens inverse de la précédente au milieu de l'intervalle

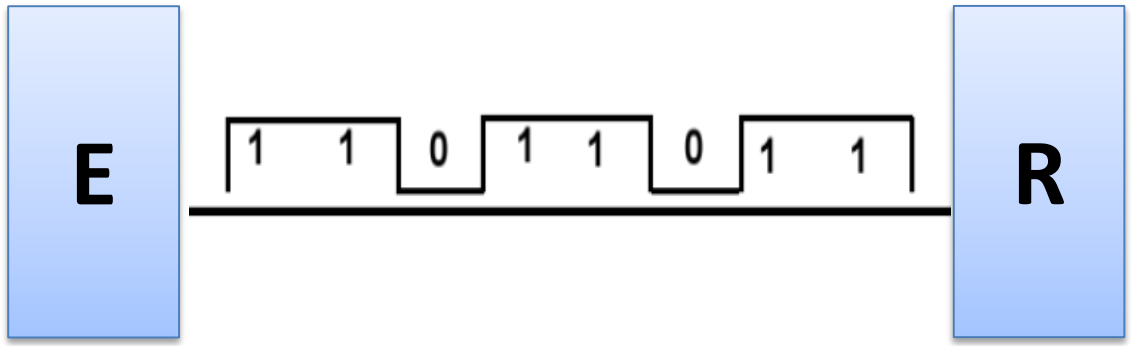
Solution d'exemple d'application 2:



3. Comment fait-on pour transmettre ces suites binaires ?

Ceci peut se faire de deux manières différentes, soit en **série** ou en **parallèle**

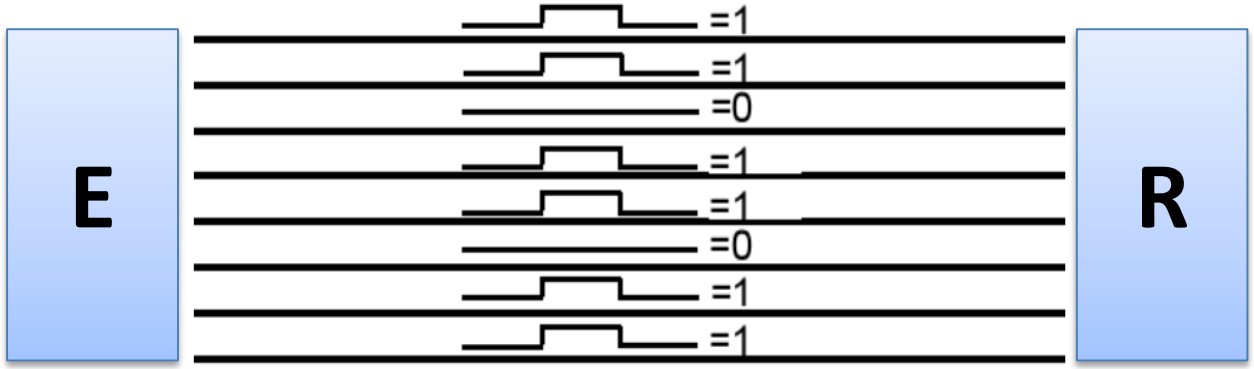
- **Liaison série**



Avantages :
- Peu de fils
- Pas Interférence à haut débit

Inconvénients:
- Lent par rapport à la liaison parallèle

- **Liaison parallèle**



Avantages :
- Rapide : plusieurs bits à la fois

Inconvénients:
- Plusieurs lignes
- Interférence à haut débit

Sans transition

1) NRZ

1: +V

0: -V

2) NRZI

1: +V,-V

0: changement

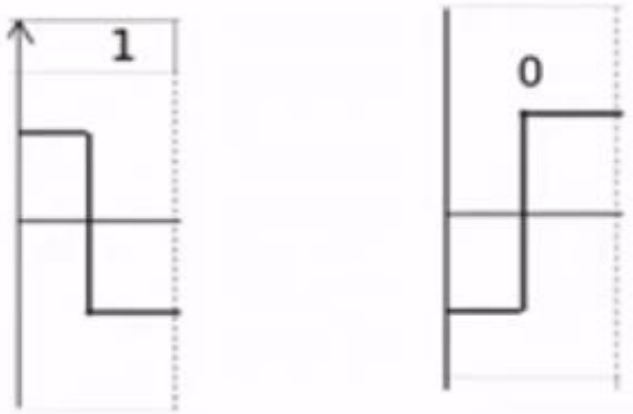
3) MLT3

1: +V,0,-V

0: changement

Avec transition

1) Manchester



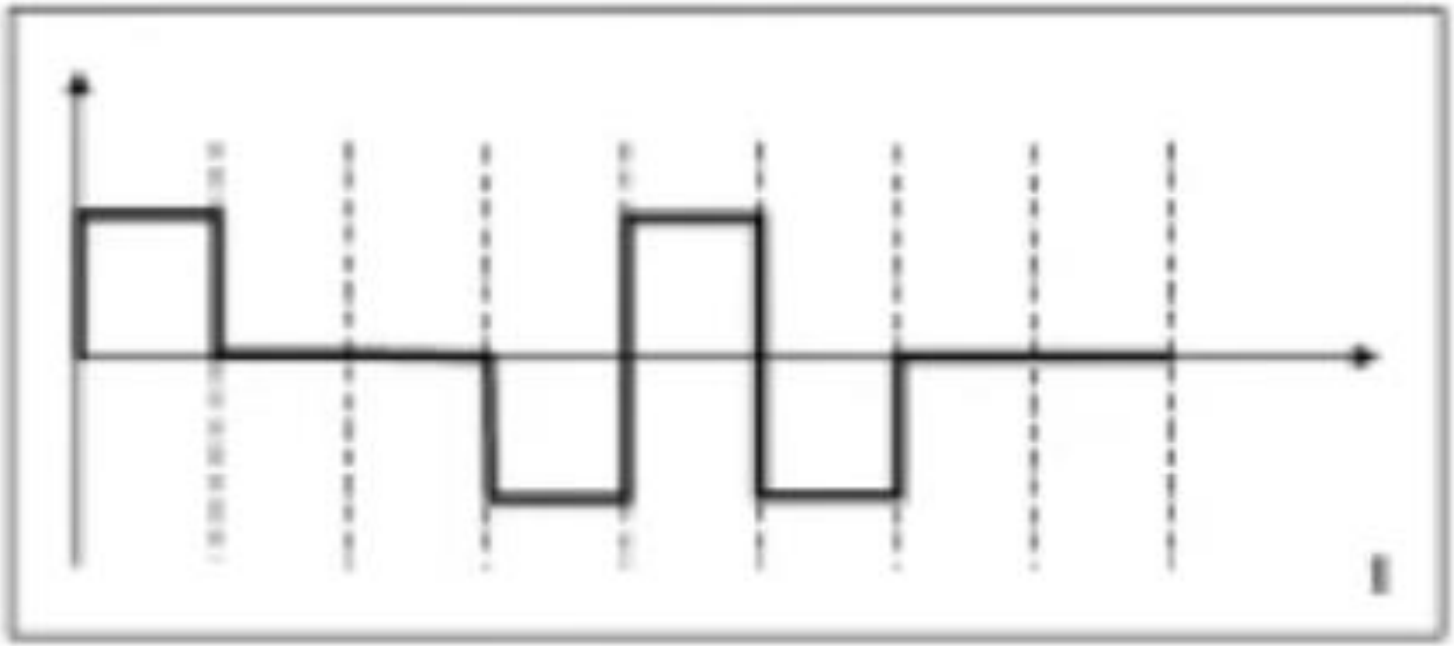
2) Manchester Diff

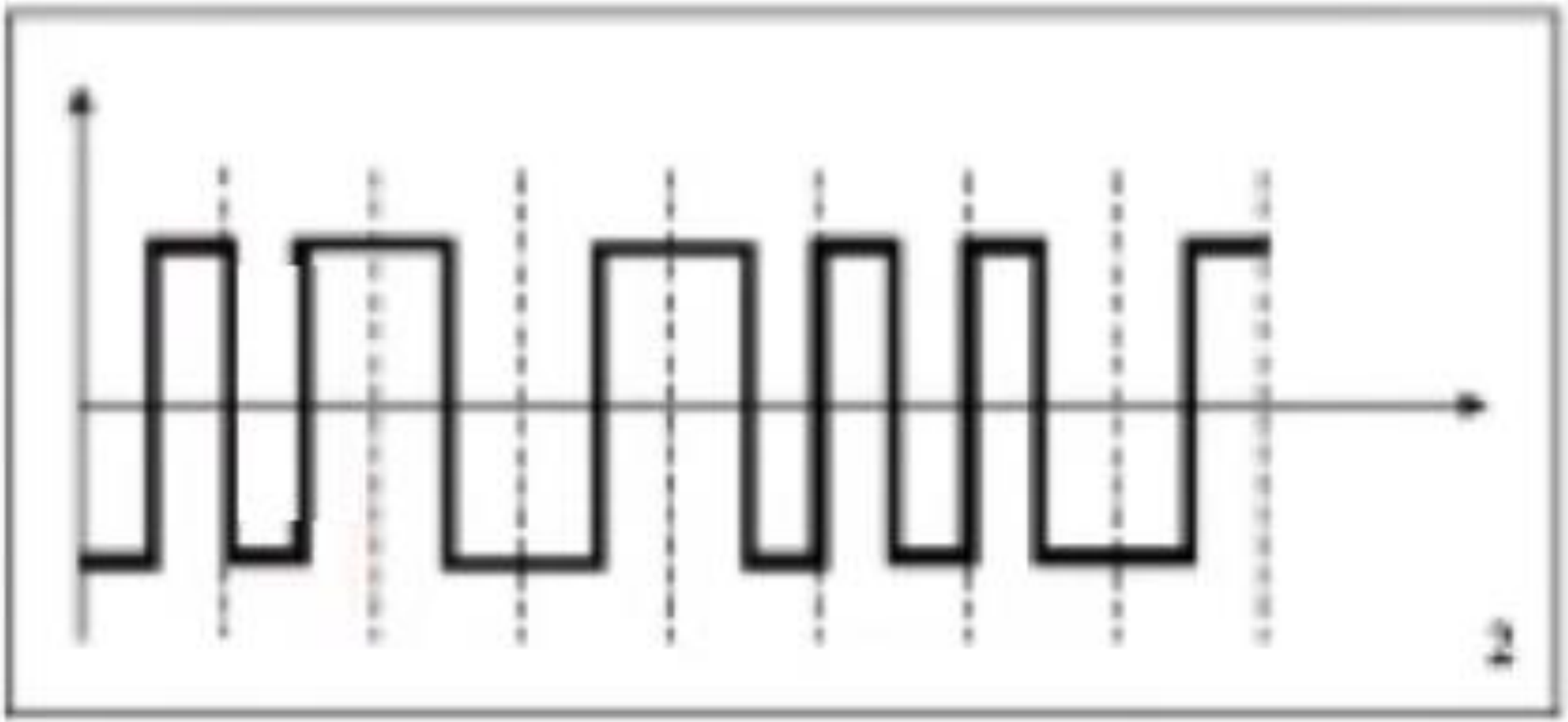
1: pas de changement du voltage de début

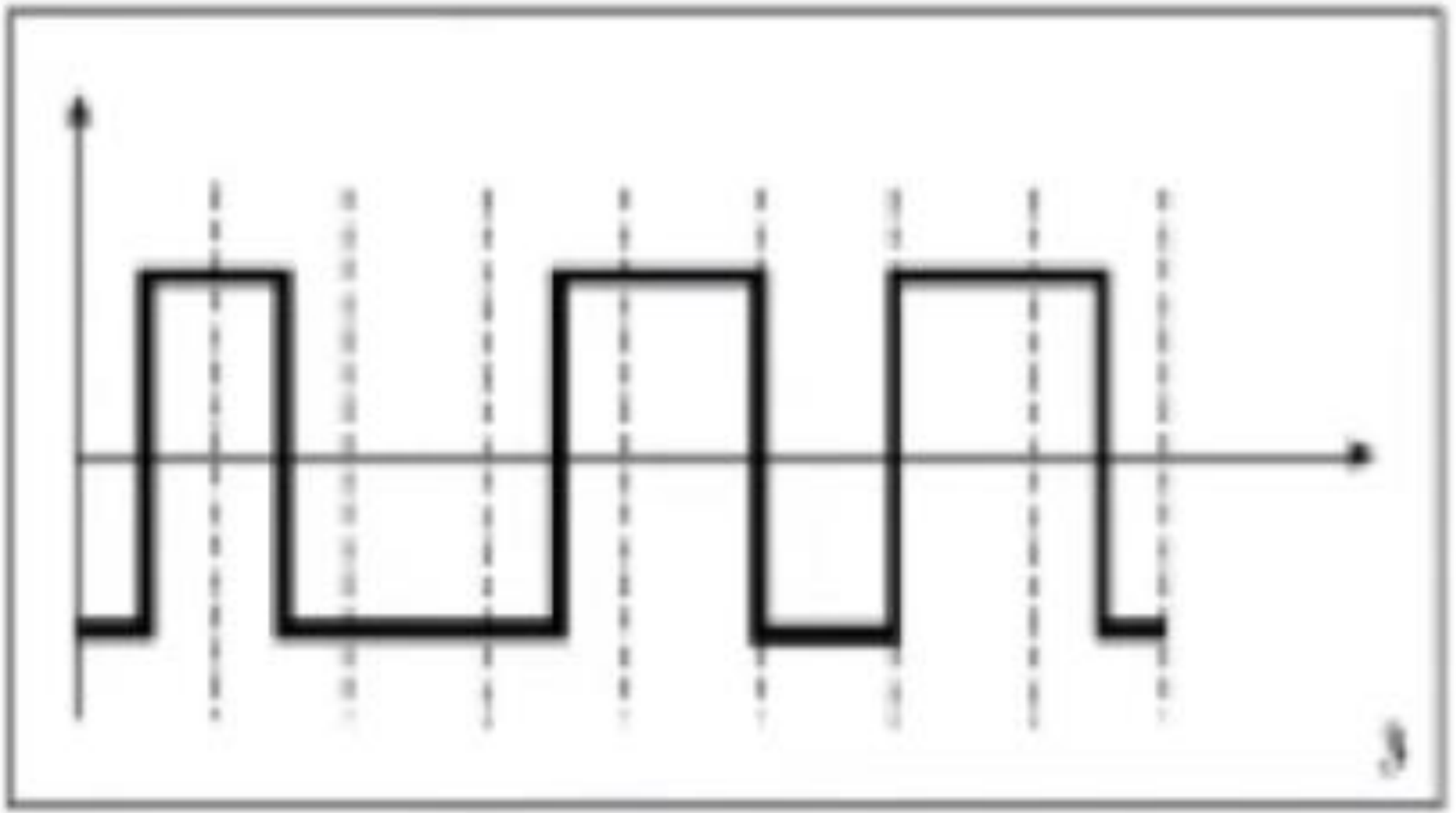
0: changement du voltage de début

Excerice

Décodez chaque séquence représentée ci-dessous en indiquant quel codage est utilisé.







Plan

1. Introduction
2. Chaîne de communication numérique
3. Codage d’une information numérique
4. **Modes de transmissions**
5. Nature de liaison

4. Modes de transmissions

En télétransmission, les équipements qui sont aux extrémités d'une liaison ne peuvent échanger des informations que s'ils utilisent les mêmes règles. On dit aussi le **même mode**.

❑ MODES SYNCHRONES

Dans le mode synchrone, les bits sont émis d'une façon régulière. Ce type de liaison utilise une ligne pour le signal à transmettre et une autre ligne (appelée horloge) pour synchroniser les échanges.

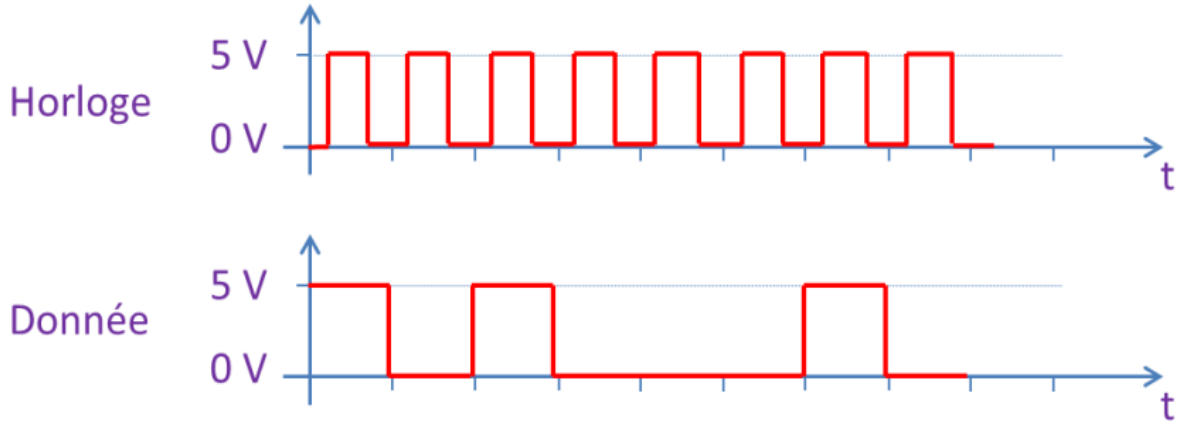
- Lorsque le signal d'horloge est actif, le récepteur lit la donnée présentée par l'émetteur.
- Lorsque le signal d'horloge est inactif, le récepteur ne lit plus la donnée, et l'émetteur peut placer une nouvelle donné sur la ligne.

❑ MODES ASYNCHRONES

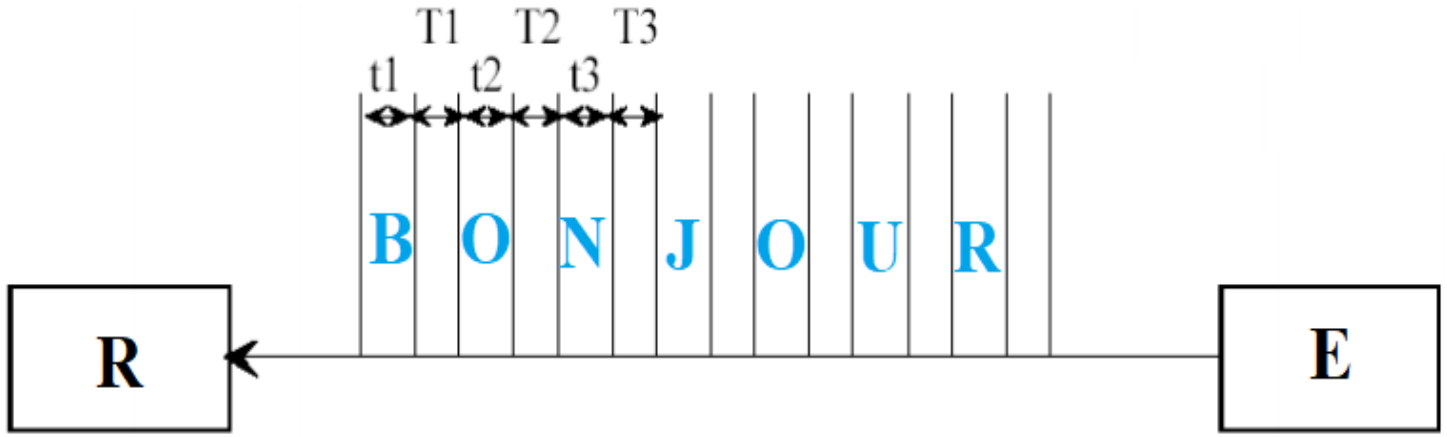
Dans une transmission asynchrone, l'horloge n'est pas transmise. Les caractères sont émis de façon **irrégulière**, comme par exemple des caractères tapés sur un clavier, l'intervalle de temps entre deux caractères est aléatoire, le début d'un caractère peut survenir à n'importe quel moment.

❑ MODES SYNCHRONES

- ❑ l'émetteur et le récepteur se mettent d'accord sur un intervalle de temps identique,
- ❑ les bits d'une donnée sont envoyés les uns derrière les autres en séquence,
- ❑ les bits sont synchronisés avec le début des intervalles de temps,



$$\begin{cases} t1 = t2 = t3 \\ T1 = T2 = T3 \end{cases}$$



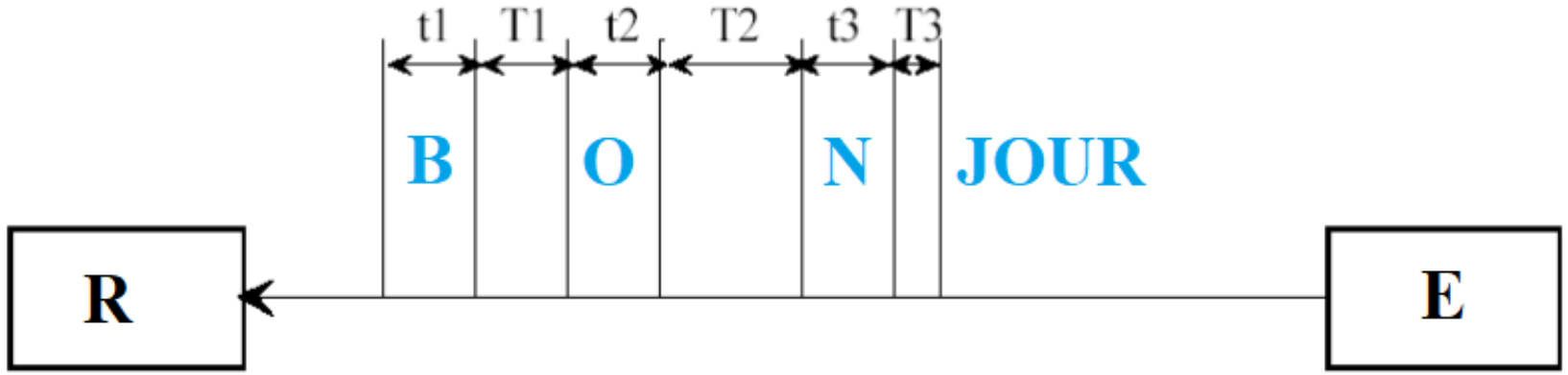
❑ MODES ASYNCHRONES

L'émetteur et le récepteur possèdent tous les deux leurs propres horloges, qui doivent être de même fréquence. Afin que l'horloge du récepteur puisse se synchroniser sur celle de l'émetteur,

Pour une transmission asynchrone:

- ❑ la source de données produit des caractères à des instants aléatoires et indépendants,
- ❑ les bits d'un caractère sont entourés de deux autres bits : bit **START** et bit **STOP**
  - Le bit **START** indique le début du caractère, déclenche l'horloge du récepteur,
  - Le bit **STOP** dont la durée peut varier (1 à 2 bits) arrête l'horloge du récepteur.

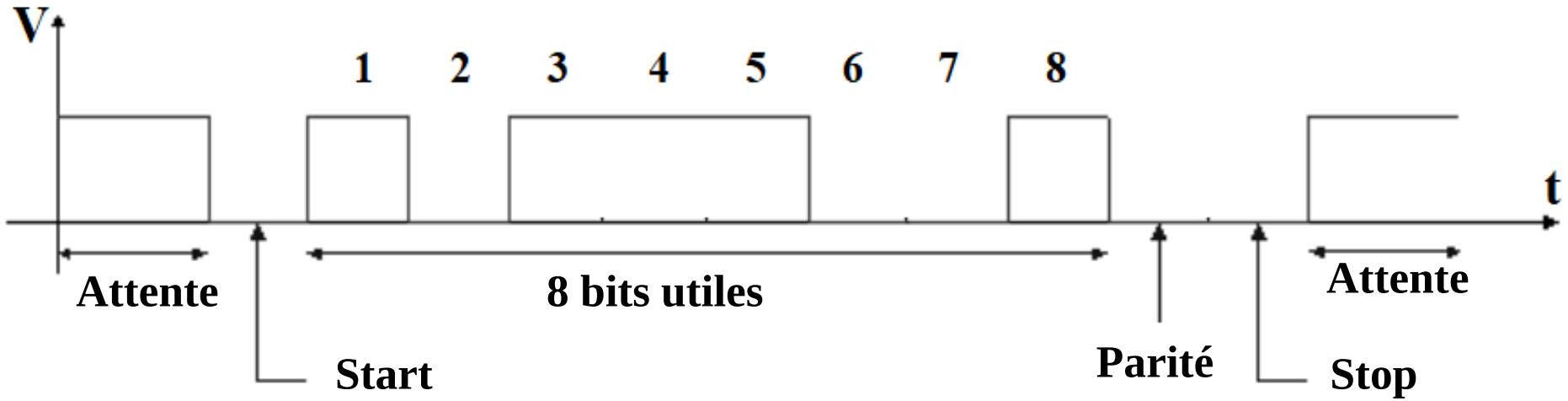
$$\begin{cases} t1 = t2 = t3 \\ T1 \neq T2 \neq T3 \end{cases}$$



Procédure de lecture :

1. Détection du bit de START
2. lire les bits du caractère envoyé (exemple: 8 bits de l'octet).
3. Les 8 bits sont suivis en général d'un bit de parité, puis d'un bit de stop.

Le bit de parité permet de détecter une erreur sur les bits d'information.
Le bit de stop permet de créer un intervalle de temps minimum avant d'envoyer le caractère suivant.



Rappel : en mode asynchrone, entre 2 caractères, la ligne reste en permanence à 1 ou à 0.

Plan

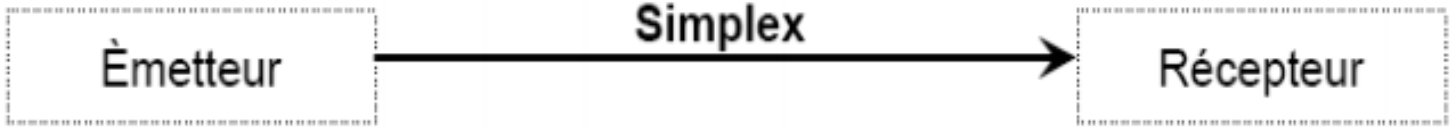
1. Introduction
2. Chaîne de communication numérique
3. Codage d’une information numérique
4. Modes de transmissions
5. **Nature de liaison**

5. Nature de liaison

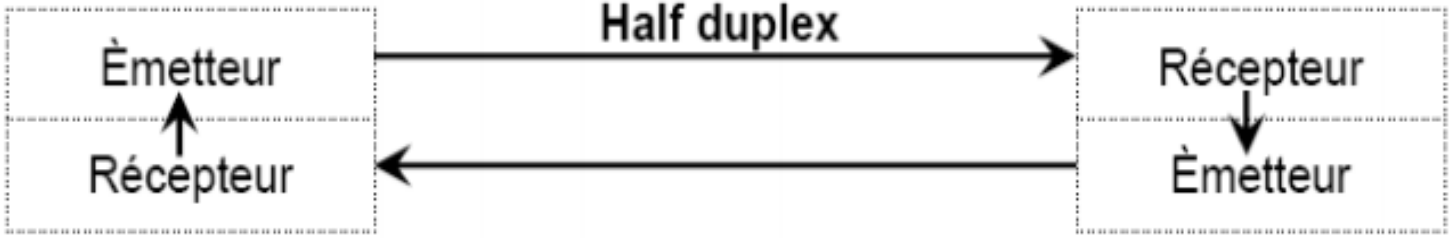
Pour transmission entre deux points, la plupart du temps, il faut traiter un dialogue et non un monologue. Il faut donc une convention pour fixer le sens de la transmission.

On trouvera 3 cas :

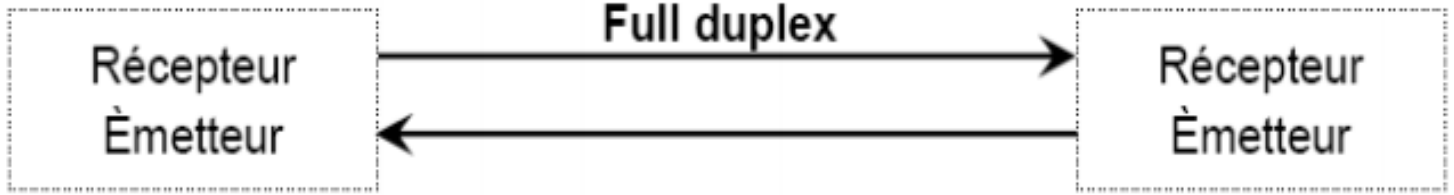
- Dans un seul sens : **Liaison simplex.**



- Dans les 2 sens non simultanément : **Liaison half duplex**.



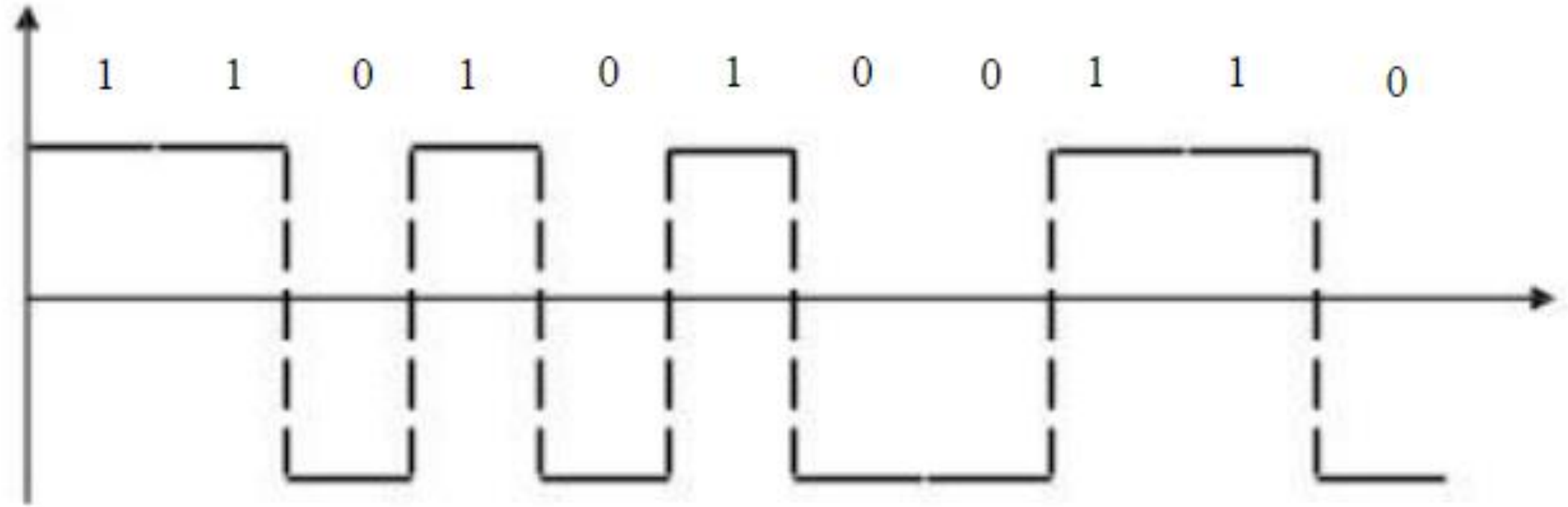
- Simultanément dans les deux sens : **Liaison full duplex** .



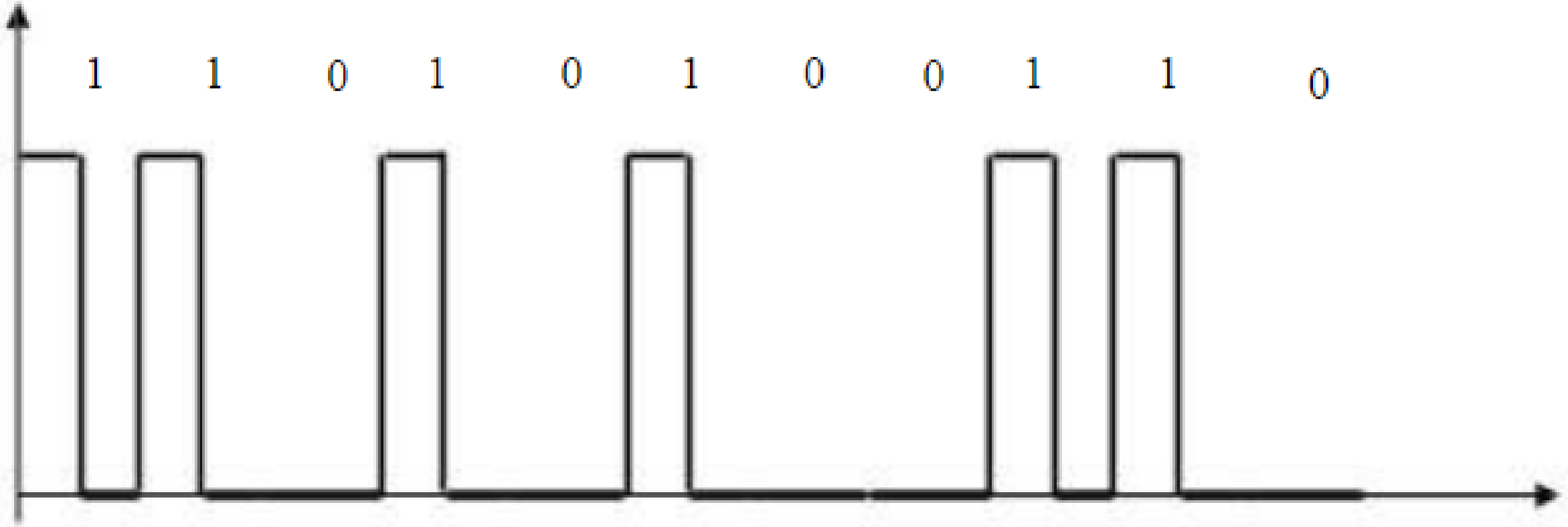
Exercice 1

Tracer le signal représentant l'information binaire 11010100110 en utilisant les différents codes vus au cours.

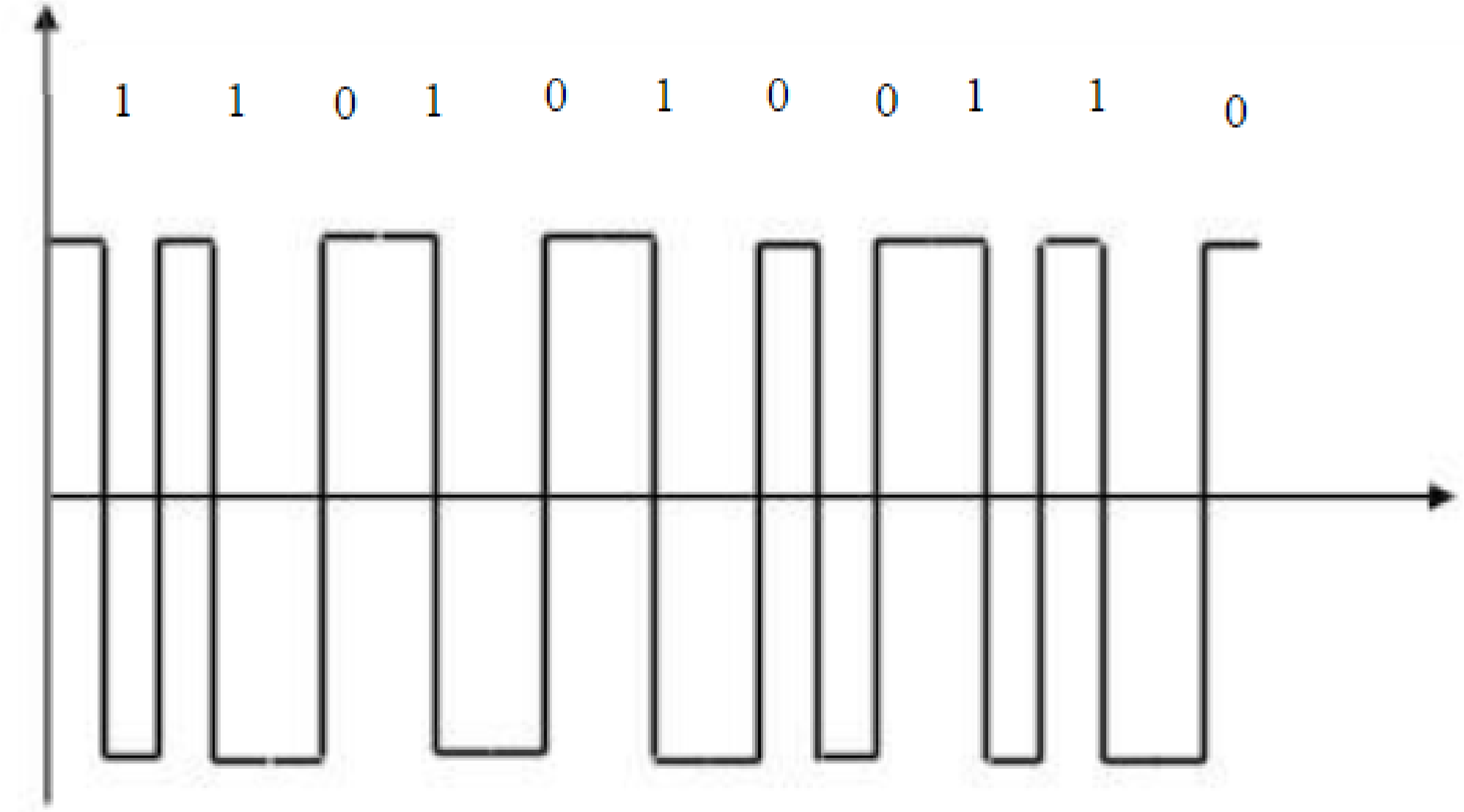
NRZ :



RZ :



Manchester :



FIN