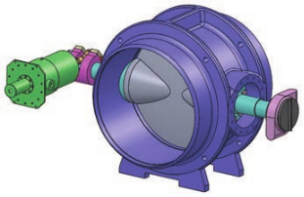
TD2 : Logique Combinatoire

1 Capteur de position utilisant le code Gray



La vanne de fermeture d'une centrale hydraulique peut pivoter de 90^o (position 0^o lorsque la conduite d'arrivée d'eau est totalement fermée, position 90^o lorsque la vanne est ouverte). Pour connaître, de façon précise et sûre, le position de cette vanne, E.D.F. l'a équipée d'un codeur absolu (sur 360^o) à code GRAY, et d'un capteur numérique à 12 bits.

Q1 Déterminer la résolution angulaire de ce système de mesure.

Q2 Pour des raisons budgétaires, votre système d'acquisition ne possède que huit voies d'acquisition. Que proposez-vous comme méthode pour obtenir la lecture de la position de la vanne la plus précise possible ?

Q3 Quelle sera la précision obtenue ?

2 Registre

Un registre est un espace mémoire permettant de stocker un mot binaire de la taille du bus et de l'UAL. Sur les microcontrôleurs 8 bits, les registres ont une largeur de 8 bits.

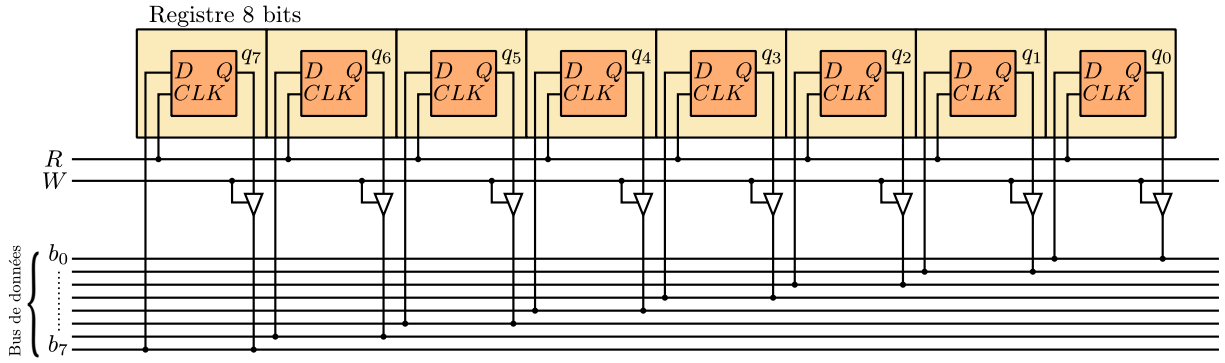


FIGURE 1 – Registre 8 bits connecté au bus de données

Les registres sont reliés au bus de données, normalement par l'intermédiaire d'un multiplexeur/démultiplexeur (voir exercice suivant) qui connecte le bon registre au bus en fonction de l'adresse fournie. Pour des raisons de simplification, la figure 1 représente le registre sélectionné par l'adresse directement connecté au bus, sans représenter le multiplexeur.

On considère la mémoire initialisée à zéro au départ, puis sollicitée en écriture (mémorisation de l'information du bus) alors que le bus vaut le nombre binaire 0b00110011, puis après une phase d'attente, sollicitée en lecture (restitution de l'information mémorisée sur le bus). Le chronogramme est partiellement représenté sur le document réponse figure 2.

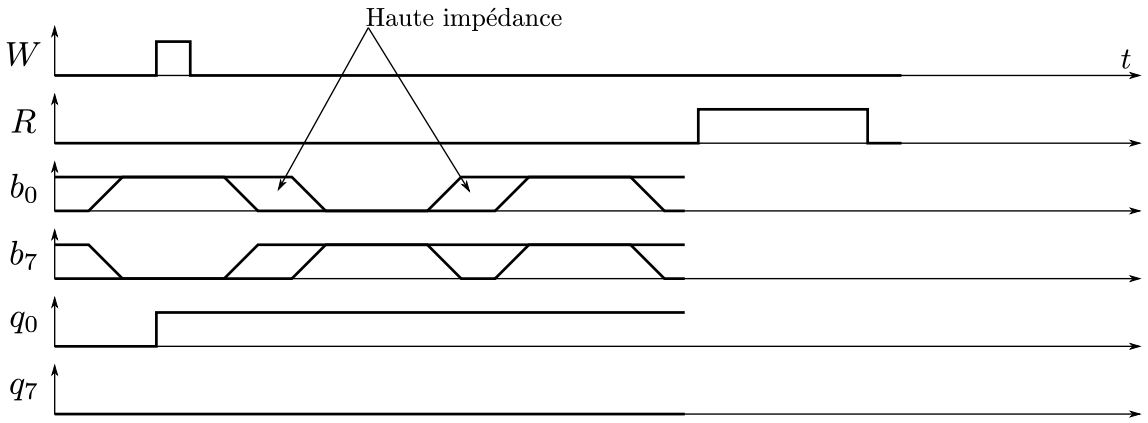


FIGURE 2 – Chronogramme d'un échange de données via le bus de données

Q4 Identifier sur le schéma les bascules D mémorisant l'information, les portes à trois états (tri-state), le bus de données et le bus de contrôle. Pourquoi le bus d'adresse n'est-il pas visible sur ce schéma ?

Le chronogramme représente une partie de l'évolution des états des variables. Seules les bits 0 et 7 sont représentés, les autres évoluant de façon similaire. Les conducteurs du bus évoluent entre des états à 0V, des états à 5V et des états intermédiaires où le potentiel n'est pas défini car aucun périphérique ne prend la main : toutes les connections sont dans un état de haute impédance.

Q5 Quelles variables représentent l'état du registre (l'information en mémoire) ?

Q6 Identifier sur le chronogramme la phase d'écriture et la phase de lecture. Vérifier que lors de l'écriture, l'état du bus est recopié en mémoire. Combien de données sont échangées sur le bus durant la phase d'attente, entre le cycle d'écriture et le cycle de lecture ?

Q7 Compléter le chronogramme en représentant les signaux en phase de lecture.

Q8 Compléter le chronogramme pour représenter une dernière phase d'écriture permettant de remettre le registre à zéro.

3 Du multiplexeur au bus de données

3.1 Multiplexeur et démultiplexeur

Un multiplexeur permet de transmettre un bit de donnée d provenant d'une des voies e_i d'un ensemble de n entrées possibles $e_0 ... e_{n-1}$ vers une destination en n'utilisant qu'un seul fil (figure 3). Le démultiplexeur réalise l'opération inverse : il transmet la donnée d d'un unique fil vers une destination s_j parmi m destinations possibles $s_0 ... s_{m-1}$. Les multiplexeurs sont généralement utilisés dans les micro-contrôleurs pour connecter quelques dizaines à quelques milliers de voies via des bus de données comme nous allons le voir ensuite. Pour des raisons de simplicité, nous nous limiterons dans l'exercice à un multiplexeur connectant 4 voies à un unique fil de transmission de donnée, puis un démultiplexeur sur 4 voies lui aussi.

On s'intéresse en premier lieu au multiplexeur. Pour définir l'entrée e_i du multiplexeur choisie, une adresse A est attribuée à chaque voie possible, sous la forme d'un mot binaire sur 2 bits $0 b a_1 a_0$. Une

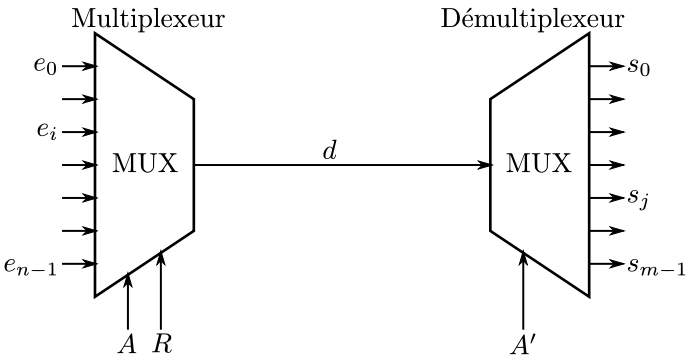
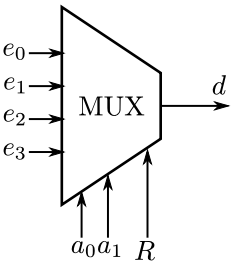


FIGURE 3 – Transmission de donnée par multiplexage

entrée supplémentaire R (read) permet d'activer la sortie d (si R=1) ou l'inhiber, c'est-à-dire la mettre en état de haute impédance (si R=0).

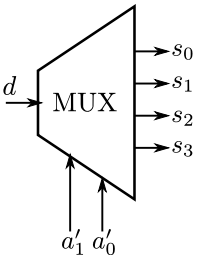
Q9 Vérifier qu'il est possible sur 2 bits d'adresse d'attribuer à chacune des 4 voies une adresse distincte et préciser les valeurs de chacune des adresses. Pour adresser une mémoire de 2048 emplacements, combien faut-il prévoir de bit d'adresse ?



Q10 Exprimer la fonction logique $d(a_1, a_0, e_3, e_2, e_1, e_0)$ puis proposer un logigramme représentant l'architecture logique du multiplexeur. Que faut-il ajouter en sortie pour prendre en compte l'entrée R ?

Q11 On souhaite programmer un PLA pour réaliser la fonction logique d du multiplexeur. Compléter le document réponse figure 4.

On s'intéresse maintenant au démultiplexeur. Les entrées du démultiplexeur sont d'une part d, connecté au fil de transmission de la donnée, et les bits d'adresse a_i. Les sorties sont les voies s_i.



Q12 Exprimer les fonctions logiques des quatre sorties s_i puis proposer un logigramme représentant l'architecture logique du démultiplexeur. On souhaite programmer un PLA pour réaliser le démultiplexeur. Compléter le document réponse figure 4.

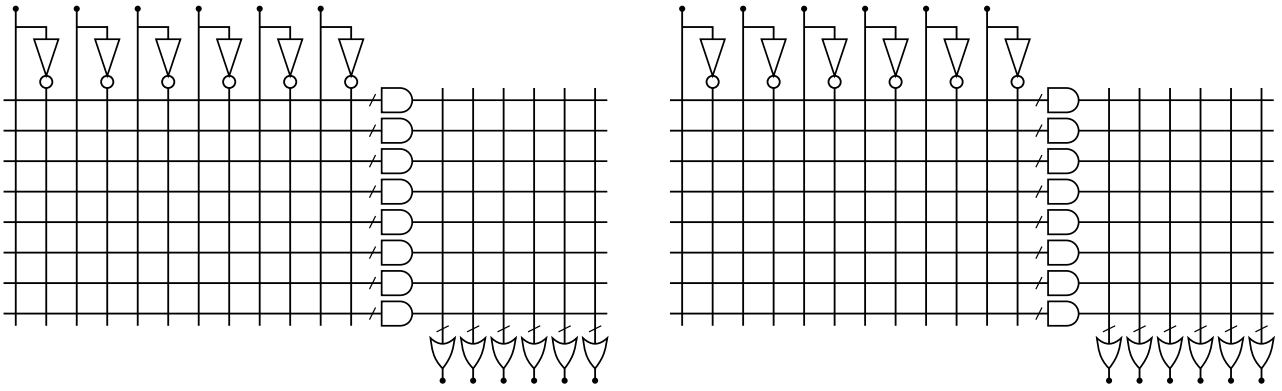
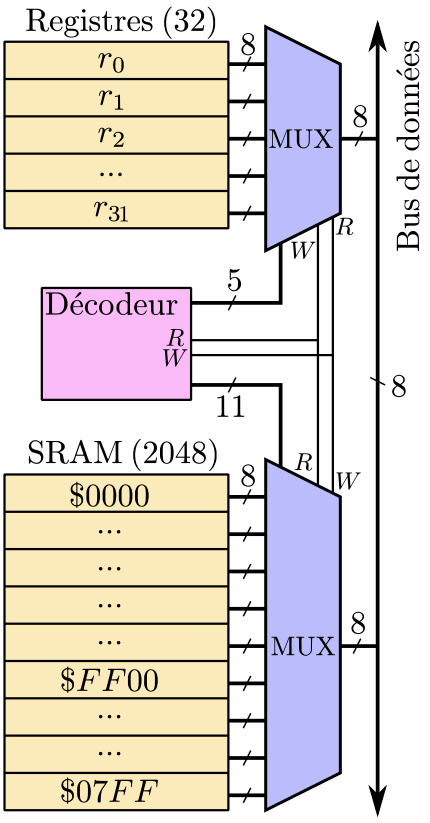


FIGURE 4 – Multiplexeur / démultiplexeur – document réponse

3.2 Transmission d'un octet sur un bus de données

Le bus de donnée d'un petit microcontrôleur est généralement d'une largeur de 8 bits, c'est-à-dire qu'il transmets un octet en parallèle à chaque échange. Il est consitué de 8 fils de données interconnectant les différents composants du microcontrôleur (registres, mémoire, périphériques...), auxquels s'ajoute deux fils de contrôle R et W (read et write). Des multiplexeurs/démultiplexeurs (MUX) s'interposent entre les cases mémoire et le bus, de façon à relier arbitrairement les cases mémoires au bus en fonction de l'adresse fournie par le décodeur d'instruction.



On s'intéresse à un échange classique : une donnée de 1 octet en mémoire, à une adresse $A_m = 0xFF00$, doit être transférée vers le registre r_2 de travail du microcontrôleur, puis augmentée de la valeur du registre r_1 et enfin renvoyée en mémoire à la même adresse. L'instruction en language machine s'écrit :

```
lds r2,$FF00; charge r2 avec le contenu de la mémoire $FF00
add r2,r1; ajoute r1 à r2
sts $FF00,r2; écrit le contenu de r2 dans la mémoire $FF00
```

Le transfert de l'octet dans le registre (première ligne) suit le protocole suivant :

— le décodeur place sur le bus d'adresse l'adresse $FF00 de la donnée en mémoire ;
— le fil de contrôle R est activé (lecture), ce qui conduit le multiplexeur de la mémoire à recopier le contenu de la mémoire sur le bus de données (8 fils) ;
— le registre r_2 mémorise l'état du bus de données.

Après l'addition, le transfert du registre dans la mémoire suit un protocole similaire

— le décodeur place sur le bus d'adresse l'adresse $FF00 du registre destination, et sur le bus de donnée le contenu du registre r_2 ;
— le fil de contrôle W est basculé à 1 (écriture), ce qui conduit le multiplexeur du registre à recopier le contenu de la mémoire sur le bus de données (8 fils) ;
— la mémoire mémorise l'état du bus à la bonne adresse via le démultiplexeur.

Q13 Justifier pourquoi il y a 5 bits d'adresse pour les registres et 11 bits pour la mémoires SRAM. Surligner en rouge le bus de données, en bleu le bus d'adresse et en vert le bus de contrôle.

Q14 Qui décide du sens de transfert de l'information (du registre vers la mémoire SRAM ou l'inverse) ? Par quels fils l'indique-t-il aux multiplexeurs/démultiplexeurs ?

4 Interface de communication série

Les interfaces de communication permettent aux composants numériques de communiquer entre eux ; l'exemple de la figure 5 montre une communication série entre un microcontrôleur et un actionneur intelligent. La communication série se distingue de la communication parallèle (globalement celle

décrite lors de la transmission d'un octet sur bus de données) au sens où les données (les bits d'un octet par exemple) sont transmises en série sur un seul fil, les bits se suivant à une cadence choisie à l'avance ou imposée par un signal d'horloge (transmis sur un fil supplémentaire ; c'est le cas de la communication série I2C par exemple). Elle nécessite beaucoup moins de broche qu'une communication parallèle.

On s'intéresse à la communication série UART-RS232 où la vitesse de transmission est choisie à l'avance : le récepteur et l'émetteur doivent être réglés sur la même vitesse de transmission (9600 BAUD par exemple) et génèrent eux même leur signal d'horloge permettant d'écrire et lire les bits les uns à la suite des autres sur le fil de communication. Pour une communication dans les deux sens, il faut pour ce protocole 2 fils reliés de façon croisée aux RX et TX respectifs.

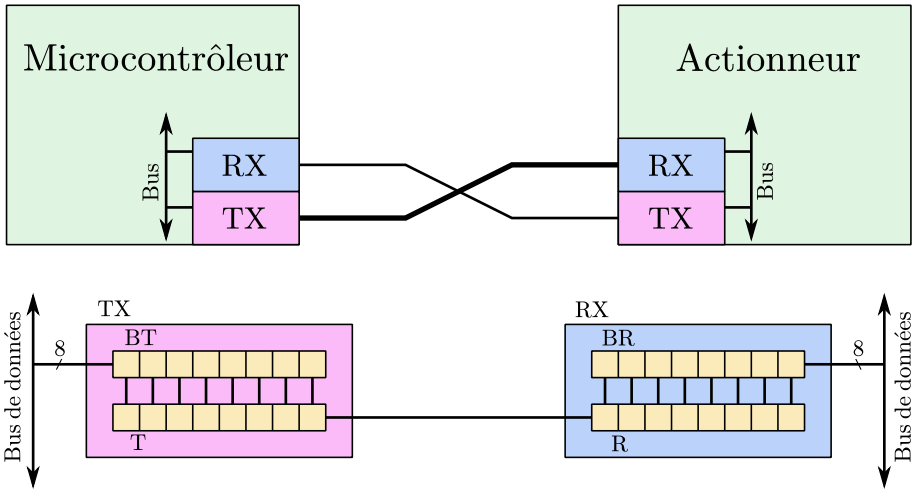


FIGURE 5 – Communication série RS232 entre deux composants.

Un module de transmission TX comporte un registre buffer BT, contenant l'octet prêt à être expédié, et connectés au bus de donnée du composant, ainsi qu'un registre de travail T qui évolue au fil du transfert de données (il se vide durant le transfert). À l'initialisation du transfert, le buffer BT est recopié dans le registre de travail T et la transmission commence. Le bus peut dès lors remettre un nouvel octet dans le buffer.

Un module de réception RX comporte un registre buffer BR, contenant le dernier octet reçu, et connectés au bus de donnée du composant, ainsi qu'un registre de travail R qui évolue au fil du transfert de données. Lorsqu'une réception est amorcée, le registre de travail R se remplit et une fois complet, il est copié dans le buffer BR qui devient disponible en lecture sur le bus de donnée, le registre de travail R étant lui disponible pour une nouvelle réception.

On cherche à transmettre l'octet de donnée 0b10011010 à travers la liaison série. On admet que les signaux CLK de part et d'autre sont synchronisés, et que Tt=Tr.

Q15 Identifier les bascules D utilisées pour les registres buffer et registre de travail dans chacun des modules TX et RX. Identifier le bus de données de l'émetteur et du récepteur, et le fil de communication série.

Q16 Compléter le chronogramme figure 6 coté récepteur et montrer que les bits de données transmis sur le câble de communication s'empilent dans le registre de travail R du récepteur.

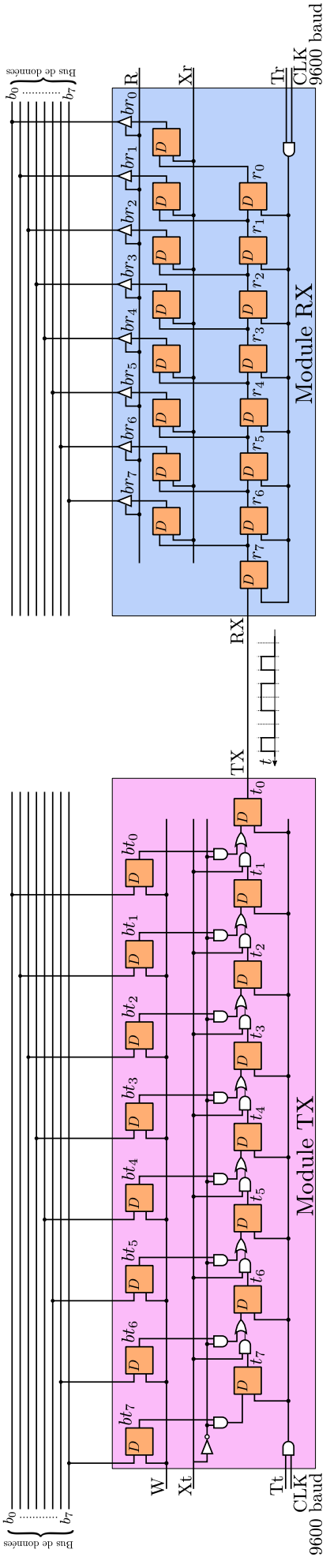
Q17 Compléter la suite du chronogramme au moment de l'impulsion Xr et déterminer quelle action est effectuée par l'impulsion Xr.

Q18 Sur le schéma de l'émetteur, colorier en rouge les portes ET qui permettent la copie parallèle du buffer BT dans le registre de travail T et en bleu les portes ET qui permettent la copie série du registre de travail sur le fil de communication. En déduire quelle valeur donner à Xt pour la copie du buffer et celle pour la transmission série.

Q19 Compléter le chronogramme coté émetteur.

Q20 À quoi servent les entrées W (coté émetteur) et R (coté récepteur) ?

Pour information, l'ajout dans la trame d'un bit de start et d'un bit de stop en début et fin (soit 10 bits transmis) permet de synchroniser les CLK en début de transmission, de calculer Tr et d'automatiser le début et la fin de la transmission, ainsi que la copie dans le buffer en fin de transmission (Xr).



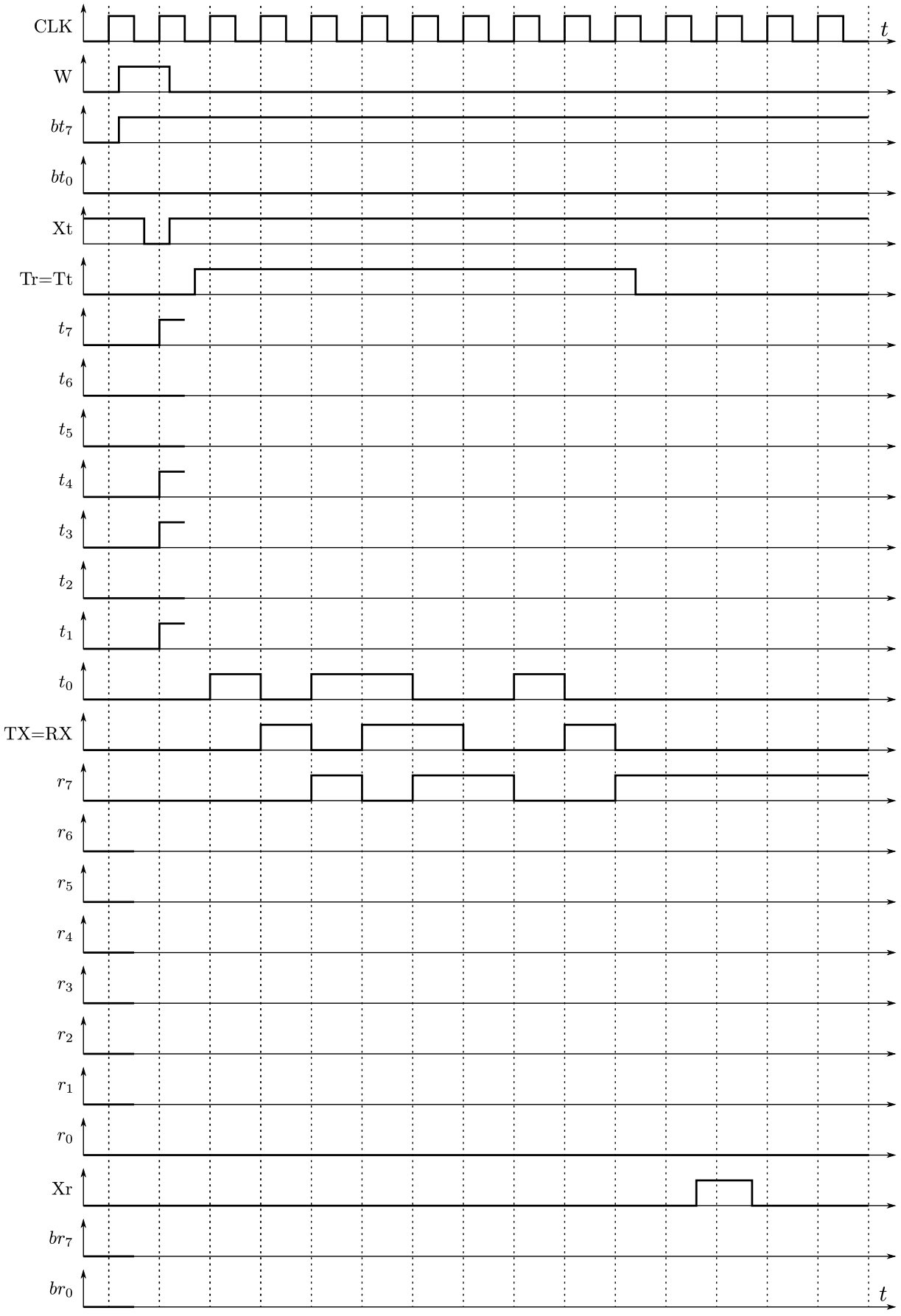


FIGURE 6 – Chronogramme d'une transmission série.