



## Programmes disponibles.

**Conventions :** Le symbole  précise que l'algorithme présente deux OPTIONS. Si **PGM** est en violet, il s'agit d'une idée personnelle pour le programme considéré.


➤ **Programmes pour les Béotiens.**


**PGM n°08 :**  Démineur "0" ou "1".


**PGM n°11 :** Écrire **SOS** en code Morse.

**PGM n°15 :** Écrire l'appel **CQ** en Morse.

**PGM n°17 :** Déminer totalement les "0".

**PGM n°28 :**  Construction d'une FRISE.

**PGM n°33 :**  Créer une clôture.

**PGM n°34 :**  Construire un mur.

**PGM n°46 :** Page "artistique" **Sinusoïdes**.


**PGM n°49 :** Recenser une population.

**PGM n°55 :** Circulariser une Orbite.

**PGM n°56 :** Calcule la monnaie à rendre.

**PGM n°57 :** Ping-pong.

**PGM n°58 :** Tir d'une flèche en Tir à l'arc.




**PGM n°45 :**  Réalise une **Dichotomie**.

**PGM n°27 :** Teste pour un PALINDROME.

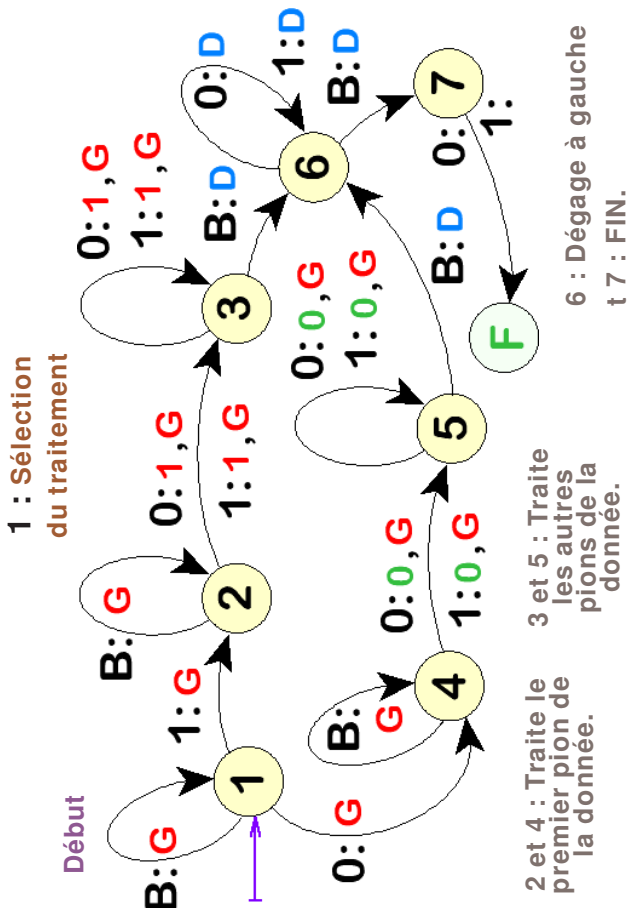
**NOTE :** *Pour imager la combinatoire explosive des algorithmes possibles, superposer les programmes 9, 19 et 27.*

## Programmes disponibles.

➤ Programmes de calculs.

**PGM n°06** : Multiplication Unaire par 2.  
**PGM n°09** :  • ADDITION Unaire.  
 • SOUSTRACTION Unaire.  
**PGM n°10** : Élever 2 à la PUISSANCE **N**.  
**PGM n°14** : Diviser un Entier par trois.  
**PGM n°16** : Multiplier deux entiers.  
**PGM n°20** : Comparaison de deux Entiers.  
**PGM n°23** :  Incrémenter / Décrémenter  
 d'une unité une donnée codée en Binaire.  
**PGM n°24** : Calcule  $5 * N$  en binaire.  
**PGM n°25** : Calcule  $N / 5$  en binaire.  
**PGM n°35** : Calcule  $5 * N + 1$  en BINAIRE.  
**PGM n°36** : Calcule  $5 * N + 2$  en BINAIRE.  
**PGM n°37** : Calcule  $5 * N + 3$  en BINAIRE.  
**PGM n°38** : Calcule  $5 * N + 4$  en BINAIRE.  
**PGM n°39** : Calcule  $N / 3$  en BINAIRE.  
**PGM n°44** : Divise un UNAIRE en deux.  
**PGM n°47** : Addition de deux BINAIRES.  
**PGM n°48** : Soustraction de deux " " " " " ".  
**PGM n°53** : PGCD de deux UNAIREs.  
**PGM n°63** : Division euclidienne de **A** par **B**.  
**PGM n°64** : Calcule  $3 * N$  en BINAIRE pur.  
**PGM n°65** :  ( $3 * N$ ) + 1 ou +2 " " " " " "

## Programme utilisateur n°1.



## Programme utilisateur **TEST 2.**

Utilitaire de servitude développé pour dépanner la machine présentant un aléa aléatoire et une "non conformité Turing". La sélection se fait avec le pion situé sous la tête de Lecture/Écriture :

**"1"** : Saut en Tr2. Puis, quel que soit l'état du BIT saut en Tr10, la TRANSITION qui Écrit un **"B"**, déplace à **G**auche et **F**in.

**BUT :** Tester rapidement si une ligne de **Fin** accepte des actions Écriture & Mouvement.

"B" ou "0" : Saut en Tr4. Cette transition génère *une boucle infinie* qui dans l'ordre :

- Écrit un "0",
- Déplace à G Gauche.

**BUT :** Initialement teste en endurance pour vérifier que la LECTURE ne s'arrête pas de façon intempestive. Permet d'observer que l'écriture d'un "0" se fait bien sur les deux cames et qu'il n'y a pas d'arrêt prématuré. Il est recommandé de placer entièrement le plateau à "B" pour voir les BITS qui ont été traités et si le carrousel a fait un tour complet. Rien n'interdit en AUTO de placer manuellement un pion à "B", à "0" ou à "1".

### Programmes disponibles.

#### > Logique de microcontrôleurs.

- PGM n°01** : Écriture de "1" ou de "0".  
**PGM n°02** : • SHIFT à Droite de "1".  
 • Tête L/E placée en fin de 2<sup>ème</sup> donnée.  
**PGM n°03** : • Inverser les BITS.  
 • Chercher [01] dans une chaîne.  
**PGM n°21** : Permutation circulaire à ➡.  
**PGM n°22** : Permutation circulaire à ⬅.  
**PGM n°29** : Duplique une chaîne Binaire.  
**PGM n°30** : Simule un registre **FIFO**.  
**PGM n°41** : Compteur / Décompteur.  
**PGM n°50** : Compteur BINAIRE GRAY.  
**PGM n°51** : Décompteur BINAIRE GRAY.  
**PGM n°59** : Symétriser un BINAIRE.

#### > Programmes de transcodage.

- PGM n°07** : • GRAI vers BINAIRE.  
 • Conversion BINAIRE vers GRAI.  
**PGM n°12** : Coder un Entier vers Binaire.  
**PGM n°13** : Coder un Binaire vers Entier.  
**PGM n°42** : Déplace une donnée BINAIRE.

#### > Traitement des chaînes.

- PGM n°04** : Concaténer deux CHs de "1".  
**PGM n°60** : Séparer en deux un BINAIRE.

### Programmes disponibles.

#### > Traitement des suites.

- PGM n°40** : Créer la suite des ENTIERS.  
**PGM n°43** : Créer la suite des ENTIERS.  
**PGM n°52** : Suite des puissances de deux.  
**PGM n°61** : Construire la suite des entiers codés en BINAIRE pur.  
**PGM n°62** : Construire la suite  $U^{n+1} = U^n + 2$  sous forme UNAIRE.

#### > Programmes Spécifiques.

- PGM n°05** : Parité du nombre de "1".  
**PGM n°18** : Conjecture de SYRACUSE.  
**PGM n°19** : Suite de FIBONACCI.  
**PGM n°31** : Deux OPT : Castor Affairé.  
**PGM n°32** : Deux OPT : Castor Affairé.  
**PGM n°54** : Maximiser les cycles Horloge.  
**PGM n°26** : Vérifie si le nombre de "0" ET le nombre de "1" est PAIR dans un Binaire.  
**PGM n°66** : "Bijection dans le plan".

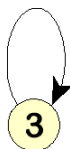
Actuellement tous les Programmes Utilisateur jusqu'au n°66 inclus et la grille de base de 561 trous représentent :

3190 trous fois deux.

### Interprétation des diagrammes.

Globalement les sites Internet qui traitent de la machine de Turing utilisent presque tous une **représentation du comportement** des algorithmes proposés **qui est le suivant** :

- Bien que ce ne soit pas un impératif absolu, par convention tout programme devrait débuter par la transition n°1 et le point d'entrée est représenté par la flèche facultative ➡.
- Les différents états de TRANSITION sont numérotés et symbolisés par : **N**
- Les TRANSITIONS sont précisées par des flèches curvilignes. Si la flèche recircule sur le même état, alors il n'y a pas de transition.



#### > Symbolisation des actions.

Elles sont codées globalement sous la forme :

- Si Lecture = **X** : faire **E**, **D**.  
**X** : Lecture de l'un des états B, 0 ou 1.  
**E** pour écrire un "B", un "0" ou un "1".  
**D** pour déplacer à **G**auche ou à **D**roite.

**F** : Déclare la **F**in du programme."

Exemples : **B: G** (Une seule action.)

**B: 0,D** (Deux actions.)

### Programme utilisateur n°1.

**Algorithme prévu comme démonstrateur d'aiguillage**, cette feuille perforée est prévue pour héberger deux logiciels binaires élémentaires. Sur la Fig.1 un premier pion de donnée en jaune sert à sélectionner le cas traité par le programme. Puis on configure une ou



plusieurs séparations de la donnée, ici représentée en rouge, par un ou des pions à l'état "B". La donnée BINAIRE est quelconque et comporte un nombre libre de BITS.

#### > Utilisation.

Placer le BIT de sélection à l'état "0" ou à l'état "1". Grand classique de base en traitements binaires, le comportement de l'algorithme est le suivant :

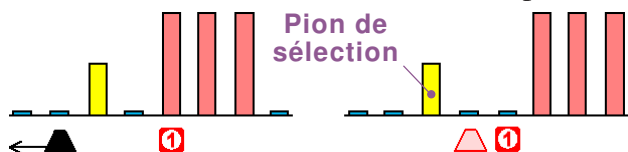
- Si le bit est à "0" l'intégralité des BITS de la donnée est forcée à l'état "0".
- Si le bit est à "1" l'intégralité des BITS de la donnée est configurée à l'état "1".
- Le processus se déroulera jusqu'à ce qu'une LECTURE détecte un état séparateur "B".

### Programme utilisateur n°2.

**D**éplacer une suite de "1" d'une case à Droite si un pion de sélection est à "0" et **mouvoir la tête de L/E à la fin d'une deuxième donnée** si le BIT d'option est à "1".

#### ➤ Utilisation pour l'option "0".

Le BIT de sélection est placé à "0". **La donnée comporte un nombre quelconque de "1"** mais attention, déjà avec 3 BITS les mouvements de la tête de L/E sont importants.



Fondamentalement ce programme réalise le SHIFT à droite d'une suite de "1".

#### ➤ Utilisation pour l'option "1".

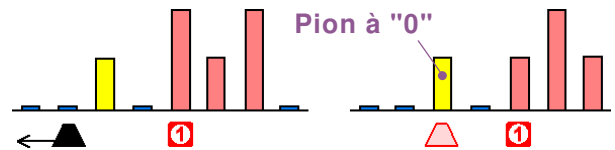
Le BIT de sélection est placé à "1". **Deux données BINAIRES** suivent le BIT d'option. **Ce programme déplace la tête L/E à la fin de la deuxième donnée.**

Fondamentalement cet algorithme simple correspond au pointage du poids faible d'une donnée BINAIRE parmi plusieurs.

... / ...

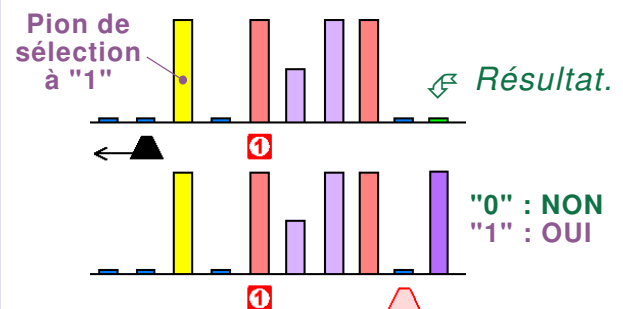
### Programme utilisateur n°3.

**A**lgorithme avec aiguillage pour deux options possibles. Banale dans son comportement, le traitement pour "0" consiste à inverser tous les BITS d'une chaîne Binaire.



#### ➤ Utilisation pour l'option "1".

Le BIT de sélection étant placé à "1" ce programme détecte si une sous-chaîne [01] est présente dans une donnée Binaire.

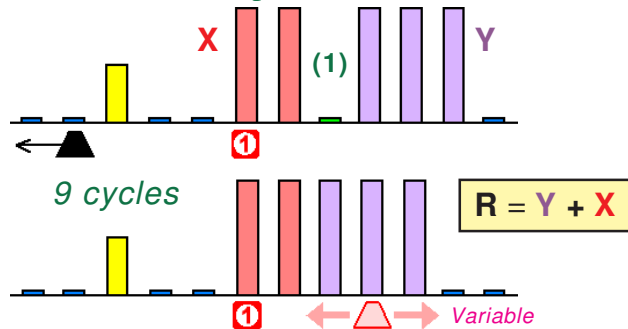


Quelle que soit la position éventuelle dans la donnée, le résultat sera écrit à droite de la valeur Binaire sous forme d'un BIT booléen.

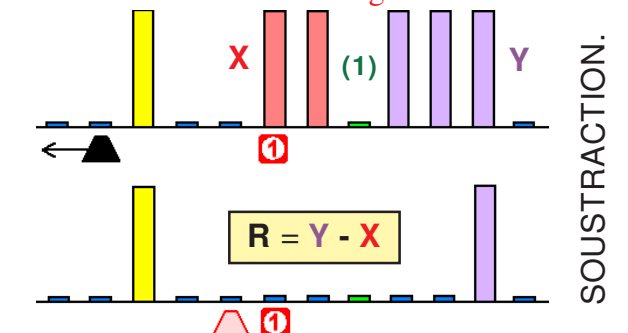
### Programme utilisateur n°9.

**A**lgorithme avec sélecteur qui pour "0" effectue une ADDITION Unaire et qui pour "1" fait une SOUSTRACTION Unaire.

(1) : UN SEUL séparateurs "B" entre X et Y.



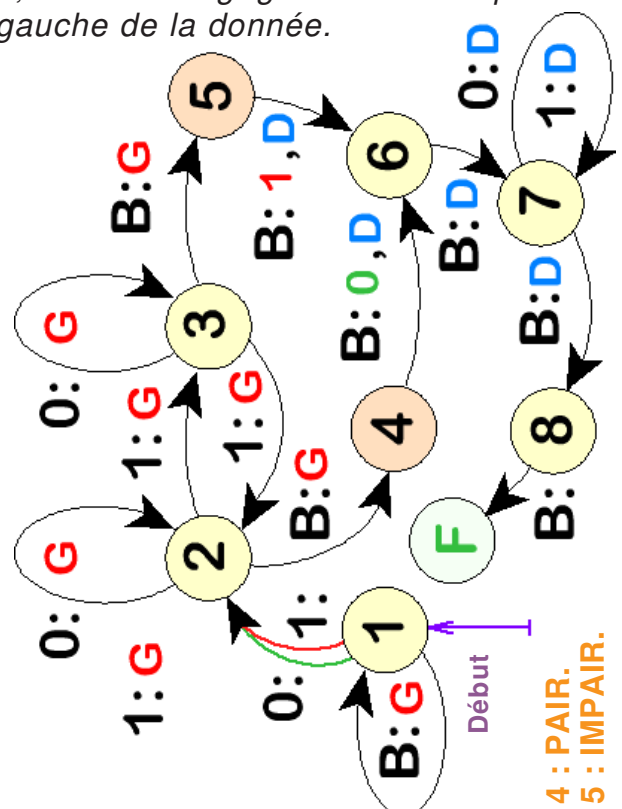
**ATTENTION** : Il faut X Inférieur ou égal à Y.  
**ATTENTION** : Il faut X à gauche de Y.



### Programme utilisateur n°5.

1 : Cherche le premier BIT.

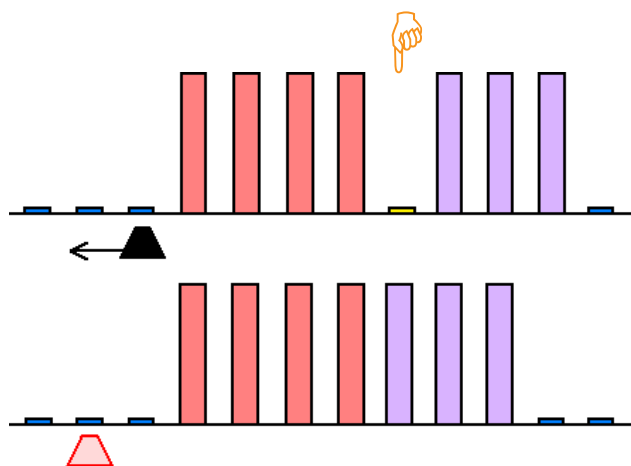
6, 7 et 8 : Dégagent de deux pions à gauche de la donnée.



### Programme utilisateur n° 4.

Bien que ressemblant à une addition de deux Entiers Unaires, ce programme effectue fondamentalement la **concaténation de deux chaînes** composées exclusivement de BITS positionnés à "1".

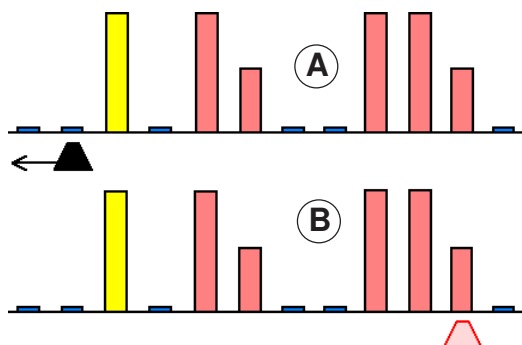
**ATTENTION** : Le séparateur entre les deux chaînes Unaires ne doit comporter qu'un seul BIT positionné à "B".



De façon banale, la tête Lecture/Écriture est positionnée à gauche des données pour faire sortir à droite le résultat.

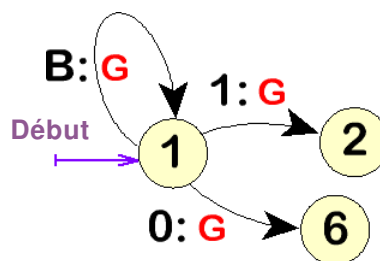
### Programme utilisateur n° 2.

Ci-dessous en **A** la configuration de départ et en **B** celle d'achèvement du traitement qui consiste à pointer la fin de la deuxième donnée BINAIRE.



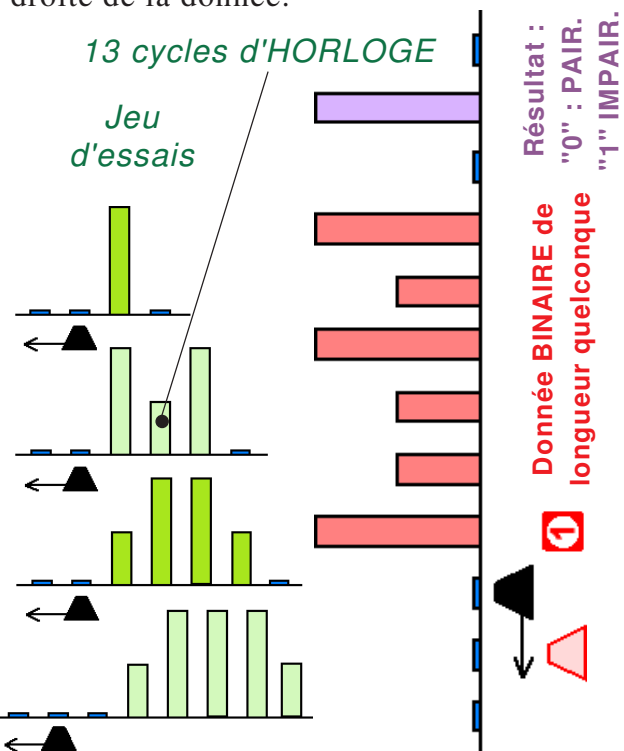
#### > Aiguillage de l'option.

Ce traitement laisse la tête de L/E sur le premier séparateur qui suit le BIT d'option.



### Programme utilisateur n° 5.

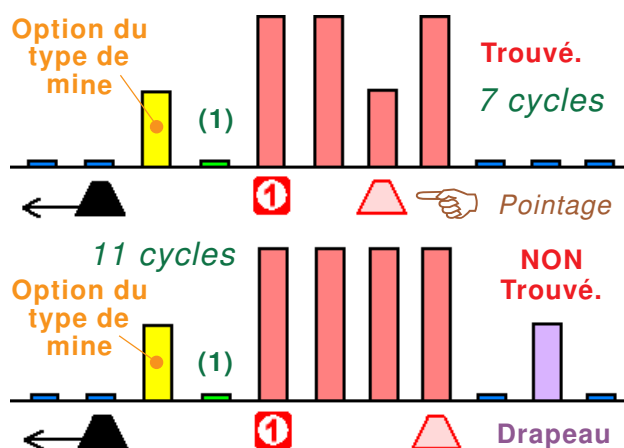
Détermine la PARITÉ du nombre de "1" dans une chaîne BINAIRE de longueur quelconque. Prévoir au moins trois "B" à droite de la donnée.



### Programme utilisateur n° 8.

**Programme de Déminage.** Un premier BIT d'aiguillage permet de choisir l'option du type de mines. Avec "0" recherche les mines de type "0", et avec "1" détecte les mines de type "1". Si aucune mine n'est trouvée dans le champ, positionne un Drapeau dont le type est celui de l'option de recherche. Si une mine est trouvée, la tête de Lecture/Écriture pointe son emplacement.

Exemples pour "0" :



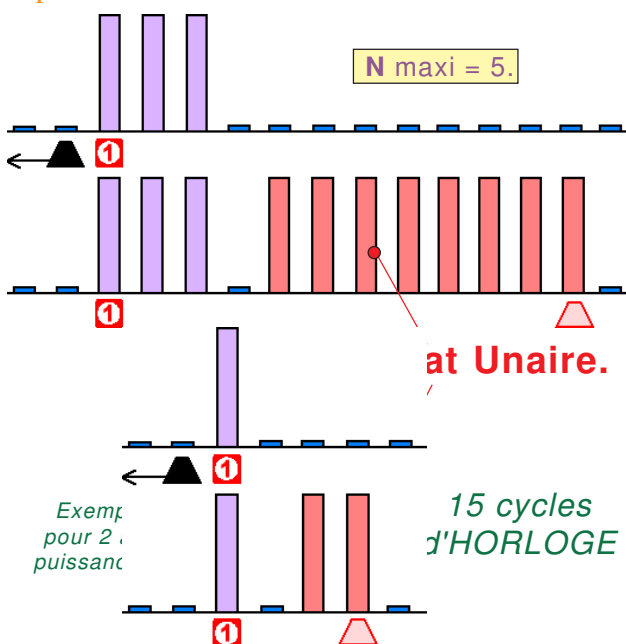
(1) : Nombre quelconque de séparateurs "B" entre le BIT d'option et le champ de mines.

... / ...

... / ...

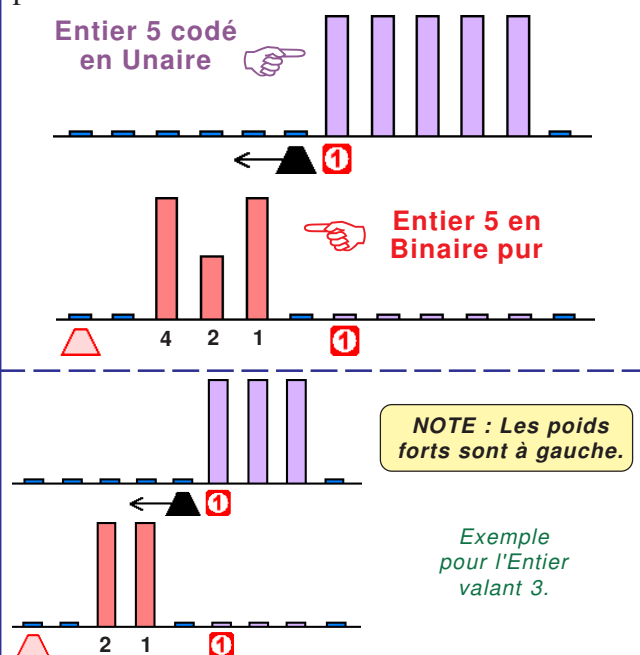
N	Binaire pur	Code GRAI
00	0000	0000
01	0001	0001
02	0010	0011
03	0011	0010
04	0100	0110
05	0101	0111
06	0110	0101
07	0111	0100
08	1000	1100
09	1001	1101
10	1010	1111
11	1011	1110
12	1100	1010
13	1101	1011
14	1110	1001
15	1111	1000

**ATTENTION** : Il faut laisser un nombre de séparateurs "**B**" suffisant à droite de **N**.



(Efface la donnée initiale.) Prévoir de l'espace de type "B" à gauche de la donnée initiale en fonction du résultat attendu.

**ATTENTION :** Déjà 65 mouvements machine pour la valeur entière 5.







### Programme utilisateur n°18.

**G**rand classique en génie informatique, cet algorithme *réalise la Conjecture de SYRACUSE* découverte par Collatz en 1930.

Prendre un nombre entier **N** :

- Si **N** est PAIR, *le diviser par 2*.
- Si **N** est IMPAIR, prendre  $(3 \times N) + 1$ .
- Recommencer avec les entiers successifs obtenus. La conjecture de Collatz prédit qu'à la fin, on obtient toujours 1.

Depuis 1930, bon nombre de mathématiciens cherchent à expliquer pourquoi cette conjecture est vraie, mais aujourd'hui personne n'y est encore parvenu. (Tout au moins en 2022.)

#### EXEMPLES :

3 ⇒ 10 ⇒ 5 ⇒ 16 ⇒ 8 ⇒ 4 ⇒ 2 ⇒ 1

4 ⇒ 2 ⇒ 1

5 ⇒ 16 ⇒ 8 ⇒ 4 ⇒ 2 ⇒ 1

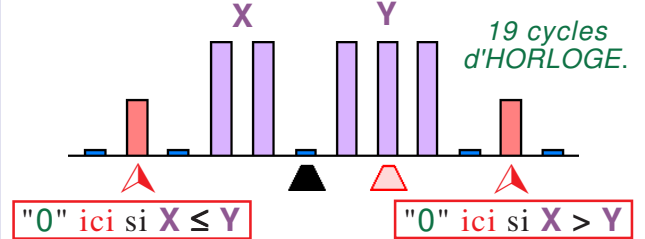
7 ⇒ 22 ⇒ 11 ⇒ 34 ⇒ 17 ⇒ 52 ⇒ 26 ⇒ 13 ⇒ 40  
⇒ 20 ⇒ 10 ⇒ 5 ⇒ 16 ⇒ 8 ⇒ 4 ⇒ 2 ⇒ 1

9 ⇒ 28 ⇒ 14 ⇒ 7 ⇒ 22 ⇒ 11 ⇒ 34 ⇒ 17  
⇒ 52 ⇒ 26 ⇒ 13 ⇒ 40 ⇒ 20 ⇒ 10 ⇒ 5 ⇒ 16  
⇒ 8 ⇒ 4 ⇒ 2 ⇒ 1 ... / ...

### Programme utilisateur n°20.

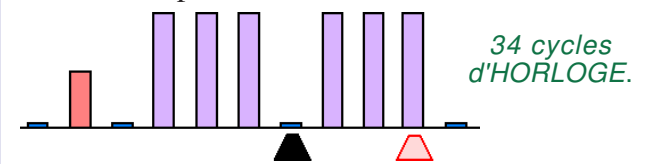
**R**éalise la comparaison entre deux Entiers codés sous forme Unaire.

Laisse la tête de Lecture/Ecriture dans une position variable à la fin du traitement. Un seul "B" sépare les données X et Y et la tête de Lecture/Ecriture doit initialement se trouver sous ce séparateur de X et de Y.



Comme précisé ci-dessus le résultat sera indiqué sous forme d'un drapeau "0" situé à gauche ou à droite des données X et Y.

Autre exemple avec  $X = Y$  :



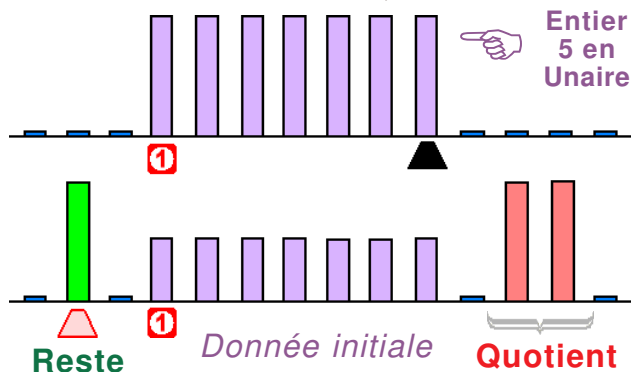
Si  $X = 3$  et  $Y = 2$  : 26 cycles d'HORLOGE.  
Si  $X = 2$  et  $Y = 4$  : 19 cycles d'HORLOGE.  
Si  $X = 4$  et  $Y = 2$  : 26 cycles d'HORLOGE.

### Programme utilisateur n°14.

**A**lgorithme qui effectue la division euclidienne par 3 d'un Entier exprimé en Unaire. Le Quotient est positionné à droite de la donnée. Les BITS de la valeur initiale sont tous remplacés par des "0".

Il faut prévoir au moins trois espaces de type "B" à gauche de la donnée pour pouvoir "dégager" la valeur du Reste et Q+2 séparateurs de type "B" à droite de la donnée pour coder la valeur qui résulte de la division entière.

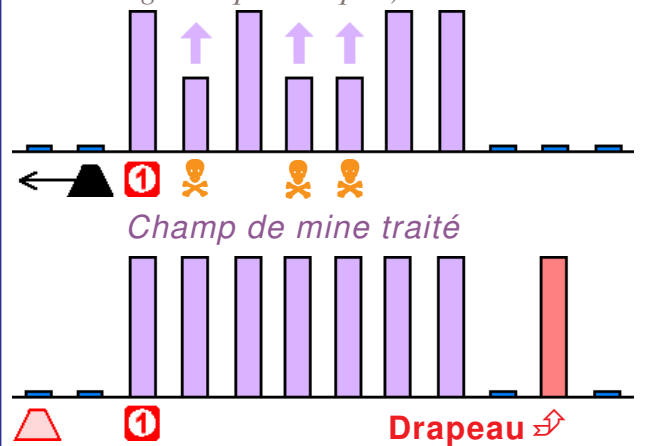
(Pour la valeur entière de 7 effectuée déjà 33 instructions sur la machine.)



**ATTENTION :** La tête de lecture/Ecriture doit se trouver à Droite de la Donnée.

### Programme utilisateur n°17.

**P**etit programme de Déminage assez voisin du programme n°8. Toutefois il ne recherche que les mines enterrées de type "0". Toutes les mines du champ contaminé sont traitées et les "trous" remplacés par des "1". Quand la totalité du champ a été sécurisé le signal par un Drapeau positionné à "1" et placé à droite de l'étendue de BITS traitée. (L'étendue du champ est de longueur quelconque.)



L'étendue traitée peut indifféremment commencer ou se terminer par une mine. La tête de L/E est dégagée à gauche.

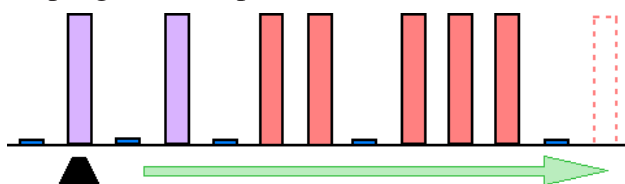
## Programme utilisateur n°19.

**A**lgorithme qui *inscrit sous forme Unaires sur la machine la suite de FIBONACCI*. Logiquement l'ensemble du barillet doit être initialisé avec des "B". Les deux premiers termes sont chacun égaux à un et la tête de Lecture/Ecriture doit se trouver sous celui de gauche.

$$U_{n+2} = U_n + U_{n+1}$$

**1 - 1**  $\Rightarrow$  2  $\Rightarrow$  3  $\Rightarrow$  5  $\Rightarrow$  8  $\Rightarrow$  13  $\Rightarrow$  21  $\Rightarrow$  34 ...

Sur la machine on ne dispose au maximum d'une étendue de 56 positions. Il y aura perte par "écrasement" des données et divergence du programme à partir de l'élément 21 inclus.



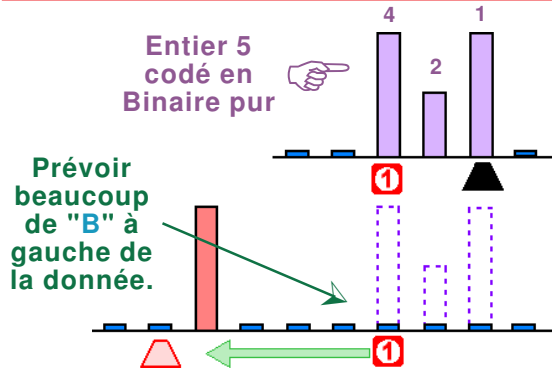
**ATTENTION :** Le nombre de cycles augmente très rapidement pour tracer les termes suivants. Par exemple quand le terme 8 est inscrit sur le plateau, la machine a déjà effectué environ 401 cycles d'Horloge.

💀 Ce programme est sans FIN. 💣

## Programme utilisateur n°18.

L'entier initial **N** est codé en **BINAIRE pur** sur la machine, les poids faibles étant à droite de façon classique. Ne pas indiquer des "0" en tête de la donnée.

**ATTENTION :** La tête de lecture/Ecriture doit être à *Droite* de la première *Donnée*.



Le programme va écrire les nombres successifs de la suite en binaire, en décalant systématiquement le codage vers la gauche. Il faut impérativement *prévoir BEAUCOUP de place libre à gauche* de la donnée initiale.

Si  $N > 7$  exige beaucoup de cycles.

Si **N** = 5 engendre 18 cycles sur la machine.

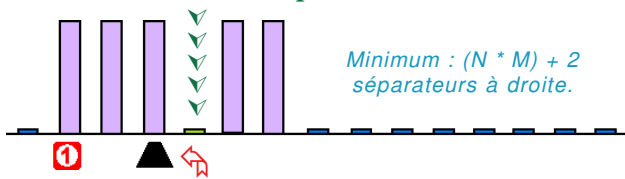
Si **N** = 7 engendre 79 cycles sur la machine.

## Programme utilisateur n°16.

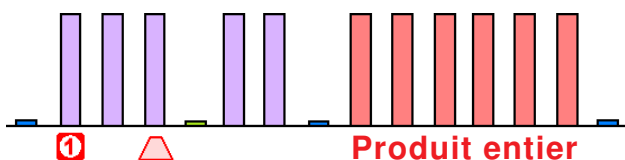
**P**rogramme de calcul qui effectue la multiplication de deux valeurs entières exprimées sous formes Unaires.

**ATTENTION :** Pour la multiplication de 3 fois 2 la machine effectue déjà 118 cycles.

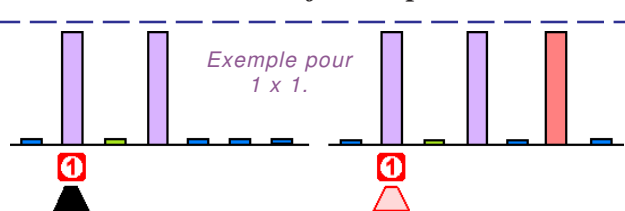
**Un seul "B" doit séparer les deux entiers.**



**ATTENTION :** La tête de lecture/Ecriture doit être à ***Droite*** de la première ***Donnée***.

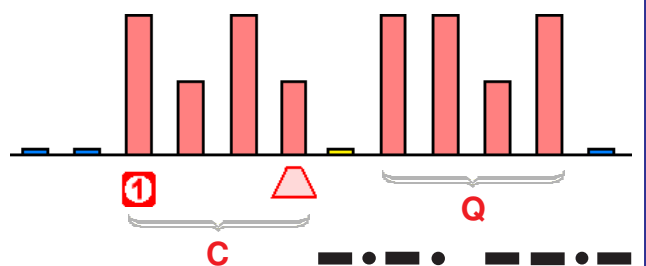


Laisse la tête L/E en *fin du premier entier*.



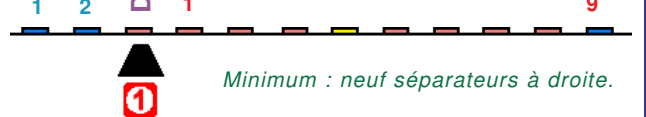
## Programme utilisateur n°15.

Logiciel très simple dont l'algorithme positionne sur le barillet le *message d'appel CQ codé en Morse*. Les points sont représentés par des "0" et les traits par des "1". L'organisation des instructions est linéaire et l'exploration des TRANSITIONS se fait dans l'ordre de la n°1 à la n°11.



Pour obtenir un résultat "dégagé" il importe de prévoir deux pions de type "B" à gauche de la tête de Lecture/Ecriture et minimum neuf séparations "B" à droite de la tête de Lecture/Ecriture.

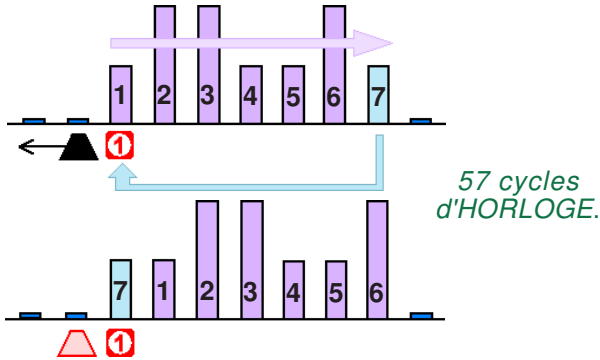
Note : Ce programme commence à écrire à partir de la position de la tête de L/E.



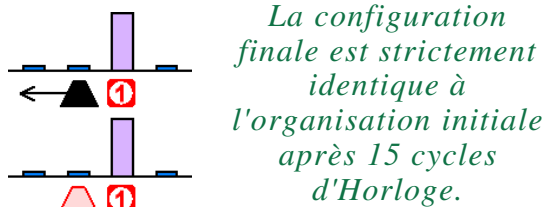


### Programme utilisateur n°21.

**R**éalise la permutation circulaire à Droite d'une donnée exprimée en Binaire pur. Laisse la tête de L/E à gauche du résultat final. La donnée binaire initiale peut présenter une taille quelconque, mais le nombre de cycles d'Horloge augmente rapidement avec le nombre de BITS.

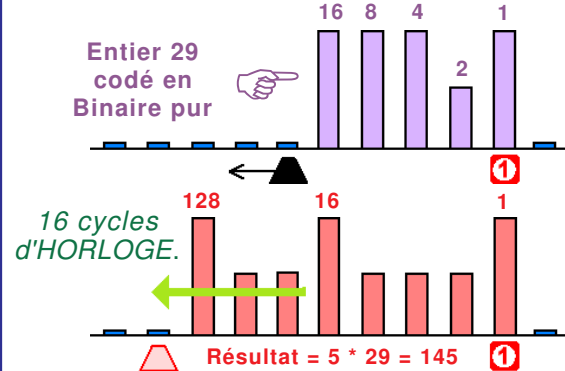


**Cas particulier : Donnée a un seul BIT :**



### Programme utilisateur n°24.

**C**alcule cinq fois l'entier N exprimé en Binaire pur. Laisse la tête de L/E à gauche du résultat final. La donnée binaire initiale peut présenter une taille quelconque, mais il faut prévoir un espace de "B" suffisant situé à gauche de la donnée pour pouvoir écrire les poids forts du résultat.



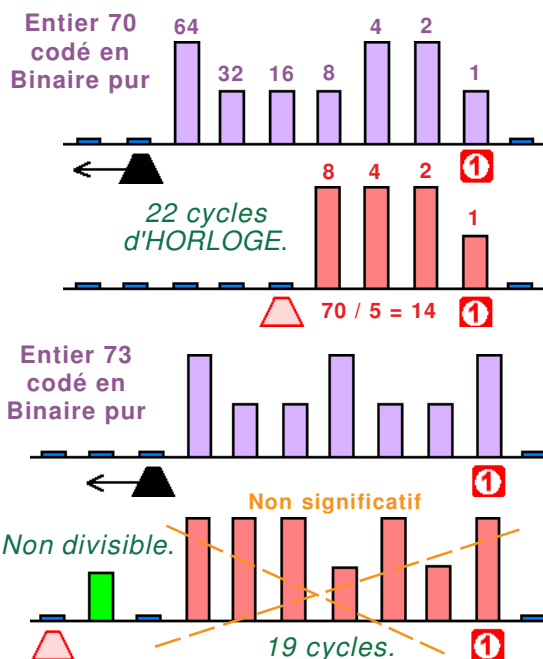
**NOTE :** Des "0" en tête, à gauche du poids fort sont autorisés mais ils seront conservés.

**AUTRES EXEMPLES :**

- N = 1, R = 5 : 7 cycles.
- N = 2, R = 10 : 9 cycles.
- N = 3, R = 15 : 9 cycles.

### Programme utilisateur n°25.

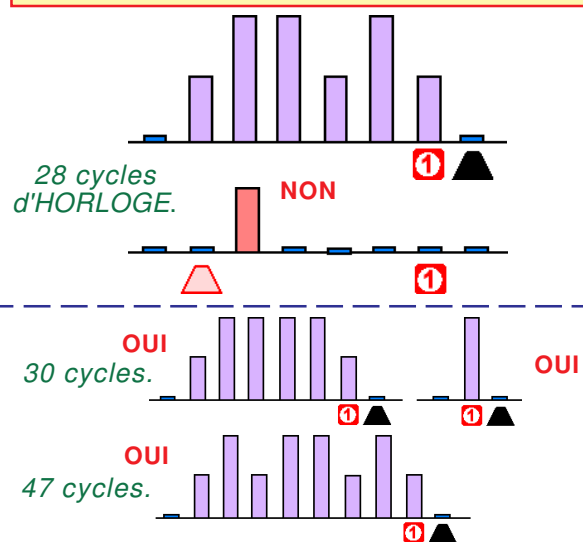
**C**alcule la division par cinq de l'Entier N exprimé en Binaire pur. Laisse la tête de L/E à gauche du résultat final. La donnée binaire initiale peut présenter une taille quelconque. Si le résultat n'est pas un entier, positionne un indicateur à "0" à gauche de la donnée initiale.



### Programme utilisateur n°27.

**A**lgorithme qui détecte si une donnée N écrite en Binaire pur constitue ou non un PALINDROME. Laisse la tête de L/E à gauche du BIT traduisant le résultat :  
 "0" : La donnée N n'est pas un palindrome.  
 "1" : La donnée N constitue un palindrome.  
 Efface la donnée initiale N.

**ATTENTION :** La tête de lecture/Ecriture doit être immédiatement à Droite de N.

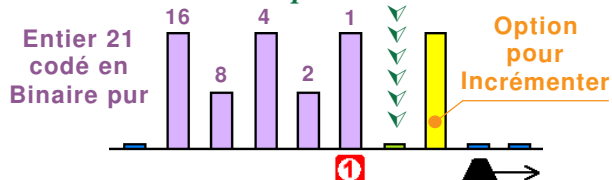


### Programme utilisateur n°23.

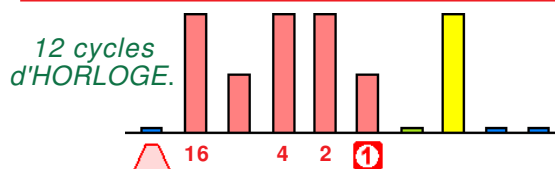
**I**ncrémenter / Décrémenter d'une unité une donnée codée en Binaire en fonction d'un BIT de sélection. Ce BIT d'option est situé à droite de la donnée Binaire séparé par un seul BIT de type "B".

- Sélecteur = "0" : **Décrém**ente de 1.
- Sélecteur = "1" : **Incrém**ente de 1.

Un seul "B" doit séparer les deux entiers.



**ATTENTION :** La tête de lecture/Ecriture doit être à **Droite** du BIT d'**Option**.



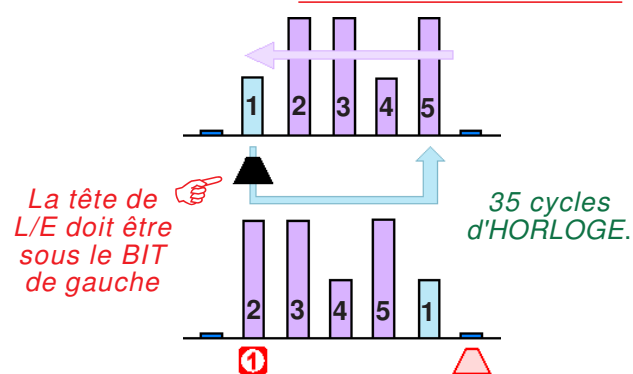
AUTRES EXEMPLES :

- 21 - 1 : 11 cycles.
- 22 + 1 : 12 cycles.

### Programme utilisateur n°22.

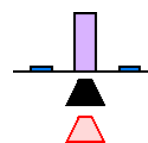
**A**nalogue au programme n°21, il effectue la permutation circulaire à Gauche d'une donnée exprimée en Binaire pur. Laisse la tête de L/E à droite du résultat final. La donnée binaire initiale peut présenter une taille quelconque, mais **le nombre de cycles d'Horloge augmente rapidement avec son nombre de BITS.**

>>> La tête L/E est sous le BIT de Gauche.



**Cas particulier :** Donnée a un seul BIT :

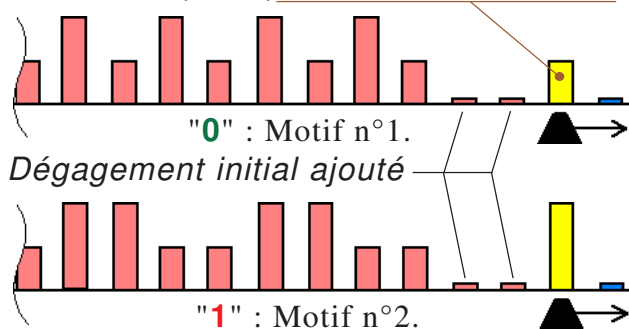
La configuration finale est strictement identique à l'organisation initiale après 10 cycles d'Horloge.



### Programme utilisateur n°28.

**P**articulièrement bien adapté pour effectuer sur la machine des test d'endurance, ce programme qui propose deux options se contente de **construire une frise sans FIN sur le barillet**. Durant son déroulement le plateau effectue sans interruption des rotations à Droite pour "dégager" le résultat. **L'aiguillage se produit sur le premier BIT différent de "B" rencontré**. Efface le BIT de sélection et ajoute deux autres "B" pour créer un dégagement initial en tête de la frise.

Sélection par le premier NON "B" rencontré.

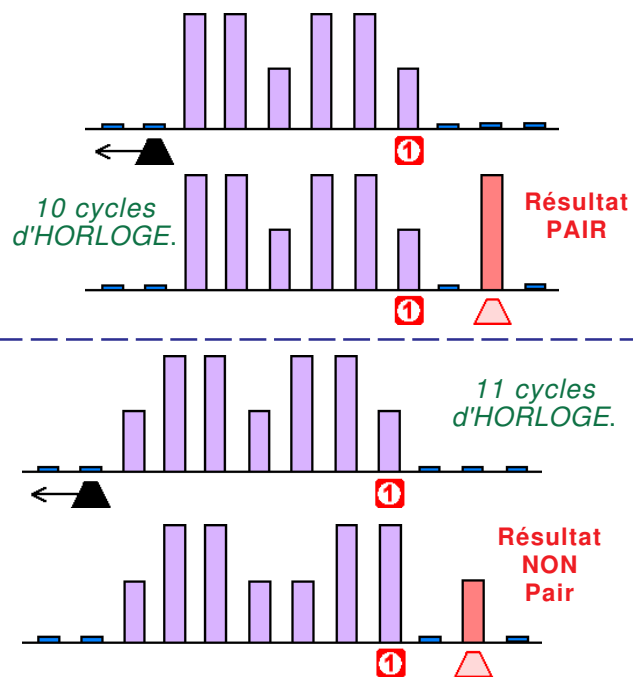


Ce programme fonctionne en "aveugle" et construit la séquence quel que soit l'état actuel des pions sur le barillet.

### Programme utilisateur n°26.

**V**érifie si le nombre de "0" est PAIR ET le nombre de "1" est PAIR sur une donnée N binaire. Laisse la tête de L/E à gauche du BIT traduisant le résultat :

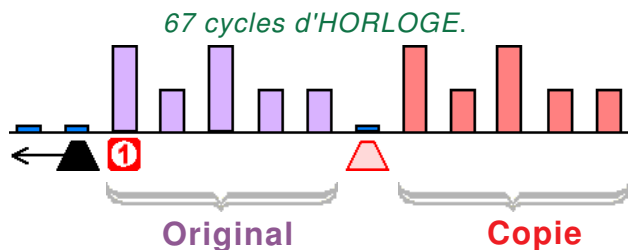
"0" : Le Nbre de "0" ou de "1" n'est pas Pair.  
"1" : Le Nbre de "0" ET de "1" est PAIR.



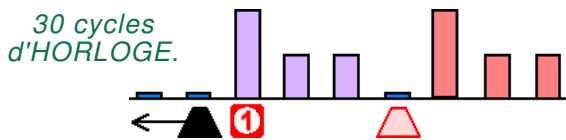
### Programme utilisateur n°29.

**T** rès orienté programmation binaire de microcontrôleurs, *ce logiciel duplique une chaîne de caractères* de longueur quelconque *écrite Binaire pur*, opération indispensable quand on désire mémoriser une valeur sur laquelle on va effectuer des modifications propres au traitement global.

Il faut prévoir (N+5) espaces de type "B" à droite de la donnée, N étant le nombre de BITS caractérisant la chaîne à recopier.



La donnée peut totaliser jusqu'à 25 BITS mais *le nombre de cycles augmente de façon significative avec la taille de l'original.*



### Programme utilisateur n°31.

**E** xemples de Castors affairés avec deux options choisies pour leur faible nombre de cycles. La zone concernée doit contenir au moins cinq "B" à gauche du pion de sélection et au moins quatre "B" à droite. Quand le programme est activé il faut que la tête de Lecture/Écriture soit au dessus du pion de sélection. Si cette position est un "B" il sera considéré comme un "0".

OPTION	États	Symboles	Cycles	Score
"0"	2	2	6 + 2	4
"1"	3	2	21 + 2	5

(Le +2 dans Cycles est issu de l'OPTION.)

Prévoir cinq à six "B" à gauche du BIT d'OPTION et quatre "B" à droite.

Déterminer un Castor Affairé est un problème non résolvable algorithmiquement. Une fonction Castor Affairé à partir d'un certain Score croît plus rapidement que n'importe quelle fonction calculable. En pratique, on ne peut même pas espérer traiter un castor affairé pour un Score au-delà de 10. (Voir la fiche du PGM n°32 qui détaille ce point.)

... / ...

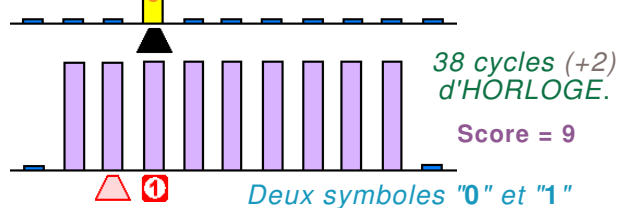
### Programme utilisateur n°32.

PGM avec options pour deux Castors affairés.

Prévoir 3 "B" à gauche et 7 à droite.

Option pour deux États

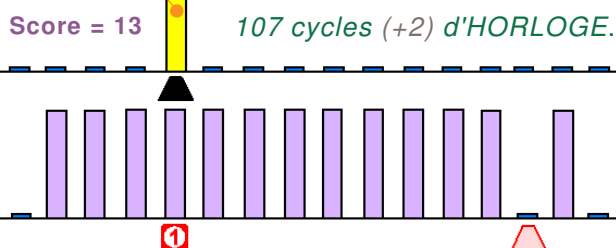
Exemples pour l'Option "0".



Prévoir 4 "B" à gauche et 11 à droite.

Option pour quatre États

Exemples pour l'Option "1".



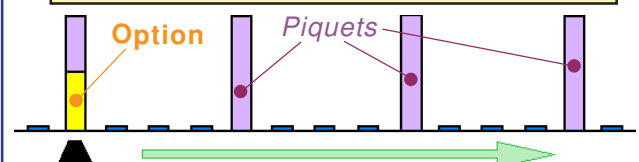
Les Pions ne sont pas consécutifs.  
Trois symboles "B", "0" et "1"

### Programme utilisateur n°33.

**C** réer un enclos quel que soit l'état des pions sur le carrousel au départ. Deux OPTIONS sont possibles en fonction de l'état du pion situé sous la tête de Lecture / Écriture. *Simule l'implantation d'une slôture par symbolisation de piquets espacés à égale distance les uns des autres.*

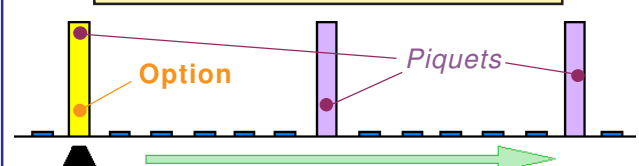
**ATTENTION : Programmes sans Fin.**

"B" ou "0" : 3 Espaces entre les piquets.



À un tour complet de barillet confirme le motif.

"1" : 5 Espaces entre les piquets.



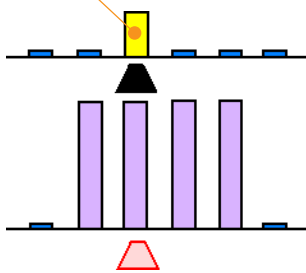
Décale d'un pion le motif à chaque tour complet du carrousel.

### Programme utilisateur n°31.

Prévoir cinq à six "B" à gauche du BIT d'OPTION et quatre "B" à droite.

Option pour deux États

Exemples pour l'Option "0".



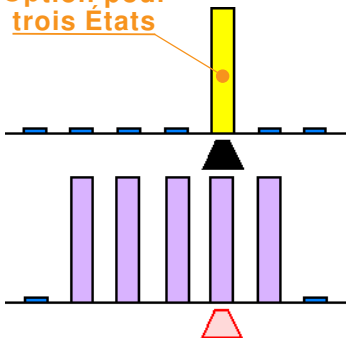
Deux symboles

6 cycles (+2) d'HORLOGE.

Score = 4

Option pour trois États

Exemples pour l'Option "1".



Deux symboles

21 cycles (+2) d'HORLOGE.

Score = 5

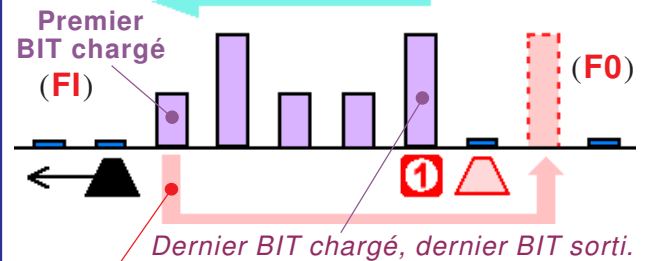
Les deux symboles sont "B" et "1"

### Programme utilisateur n°30.

Typique des opérations effectuées en binaire sur les microcontrôleurs, *ce logiciel simule le fonctionnement d'un Registre FIFO*. Ces opérations matérialisent des files d'attente en mémoire. Dans cet algorithme *le registre Binaire est supposé avoir été chargé de la Droite vers la Gauche*, le BIT d'écriture et de lecture étant à Droite. Il faut prévoir 3 à 4 "B" à droite du Registre.

**FIFO : First IN, First OUT.**

Sens supposé de chargement du Registre



Le premier BIT chargé d'initiales FI est le premier BIT sorti d'initiales FO.

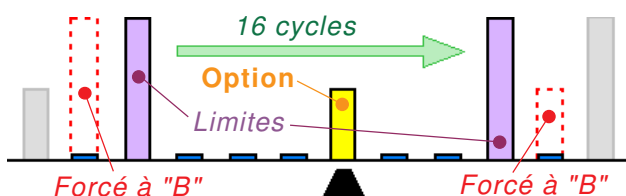
Pour simuler le fonctionnement d'un registre FIFO, quand le dernier BIT a été présenté en sortie, le BIT de transfert est effacé à "B" pour simuler sa lecture suivi de sa purge.

### Programme utilisateur n°34.

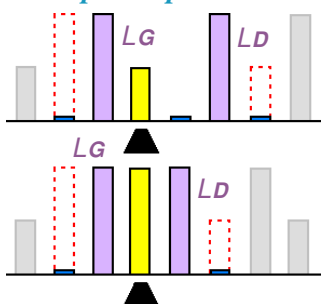
**C**onstruit un mur entre deux limites. Deux OPTIONS possibles en fonction de l'état du pion situé sous la tête de L / E :

- "B" ou "0" : Réalise de Gauche à Droite.
- "1" : Réalise de Droite à Gauche.

Force à "B" les pions de part et d'autre des limites, et termine sur le pion "effacé". À l'extérieur des Limites les états du plateau peuvent être strictement quelconques.



Bien que ce ne soit pas vraiment logique le pion d'Option peut être contre les limites :



Configuration illogique mais qui fonctionne.

### Programme utilisateur n°32.

Ce concept introduit en 1962 de *castor affaîré*, dont le nom (1) est proposé par le mathématicien hongrois Tibor Radó est l'un des 1<sup>er</sup> *exemples de fonction non calculable*. Un castor affaîré est, en théorie de la calculabilité, une machine de Turing qui maximise son activité opérationnelle comme le *nombre de pas effectués ou le nombre de symboles écrits avant son arrêt* : Son *Score*.

Ces classes de Machines de Turing doivent s'arrêter **après être lancées sur un ruban vierge** et respecter les particularités :

- La machine utilise un ruban unique,
- La longueur du ruban est illimitée,
- L'alphabet est {0, 1} avec le "0" servant servant de symbole vierge.
- La Transition dépend de deux entrées :
  - \* L'ÉTAT actuellement actif,
  - \* Le Symbole lu en début de cycle.

États	2	3	4	5	6
Cycles	6	21	107	47176870	7,4 x 10 <sup>36534</sup>

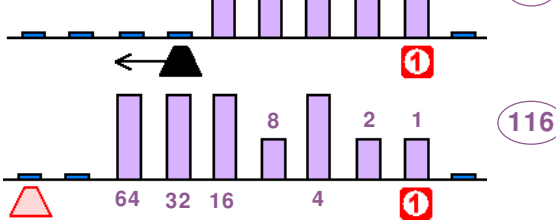
(1) "Busy beaver" : Désigne familièrement une personne besogneuse et travailleuse.

### Programme utilisateur n°35.

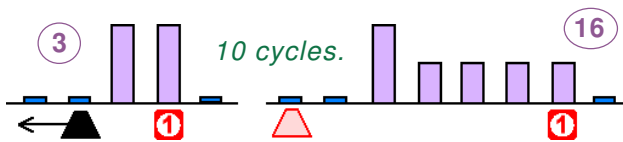
**C** *alcule  $5 * N + 1$  en BINAIRE pur.* Peut calculer des valeurs vraiment très grandes. Maximum 2 élevé à la puissance 52 soit la bagatelle de 9.007.199.254.740.991 !  
**ATTENTION :** Le nombre de cycles d'HORLOGE augmente très rapidement avec la valeur initiale proposée pour **N**.

Prévoir un nombre suffisant de "B" à Gauche de la donnée initiale **N**.

15 cycles d'HORLOGE. (23)



Par exemple si **N** = 13.421.771 le résultat sera égal à la valeur binaire de 67.108.856 soit à peine 34 BITS modifiés sur le carrousel.

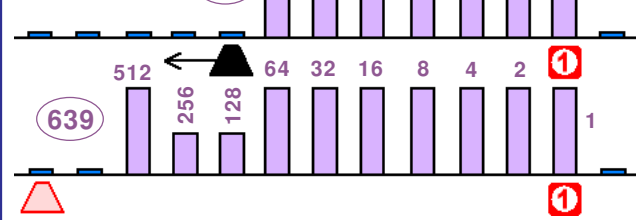


### Programme utilisateur n°38.

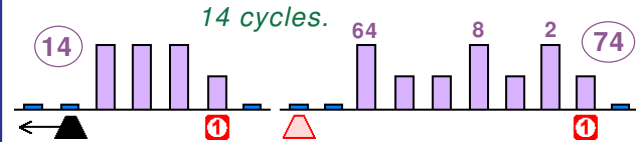
**C** *alcule  $5 * N + 4$  en BINAIRE pur.* Peut calculer des valeurs vraiment très grandes. Maximum 2 élevé à la puissance 52 soit la bagatelle de 9.007.199.254.740.991 !  
**ATTENTION :** Le nombre de cycles d'HORLOGE augmente très rapidement avec la valeur initiale proposée pour **N**.

Prévoir un nombre suffisant de "B" à Gauche de la donnée initiale **N**.

20 cycles d'HORLOGE. (127)



Par exemple si **N** = 13.421.771 le résultat sera égal à la valeur binaire de 67.108.856 soit à peine 34 BITS modifiés sur le carrousel.

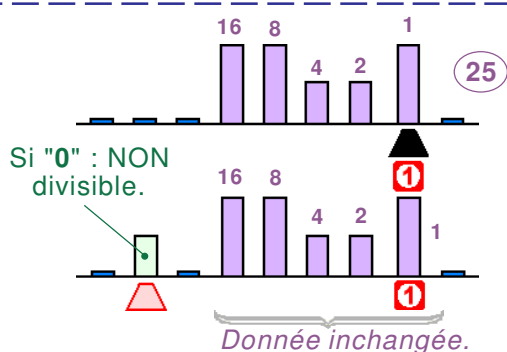
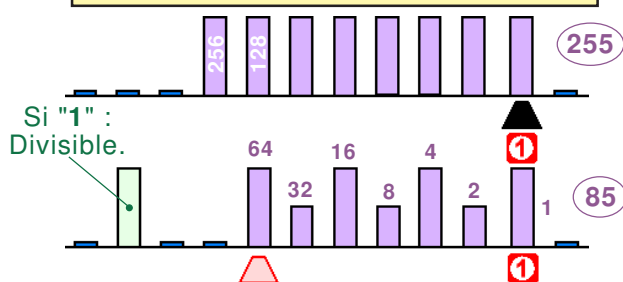


### Programme utilisateur n°39.

**E** *ffectue la division d'un nombre **N** par 3 expriméen BINAIRE pur. Teste la divisibilité et supprime les zéros en tête.*

**Note :** Il peut y avoir un ou plusieurs séparateurs entre la fin de la donnée et l'indicateur de divisibilité.

Prévoir 3 "B" à gauche et 1 à droite.



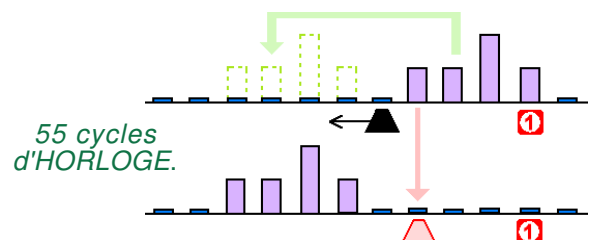
Donnée inchangée.

### Programme utilisateur n°42.

**D** *éplace une donnée BINAIRE à gauche dans la mémoire vive de la machine.* Le décalage se fait à deux BITS à gauche du poids fort de la donnée initiale qui est effacée de la mémoire. Sur un ordinateur actuel une telle donnée serait déplacée d'une cellule mémoire au format 8, 16, 32 ou 64 BITS. Sur la machine de Turing dont la mémoire est non formatée il y a optimisation de l'espace mémoire, le décalage de la transposition est minimalisé.

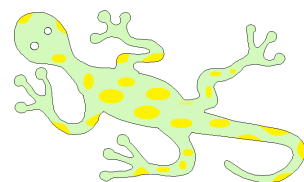
Prévoir autant de "B" à gauche de la donnée qu'elle compte de BITS pour sa définition initiale **plus trois espaces**.

Laisse la tête de L / E sur le BIT de poids fort effacé de la donnée initiale.



**ATTENTION :** Le nombre de cycles pour l'HORLOGE augmente très rapidement avec le nombre de BITS de la donnée BINAIRE.



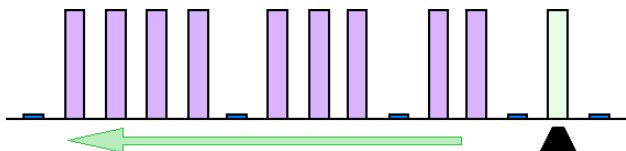


### Programme utilisateur n°43.

**C**onstruit la suite des entiers naturels codés en **UNAIRE**. La machine recopie à droite l'élément  $n$  et ajoute un pion à "1".

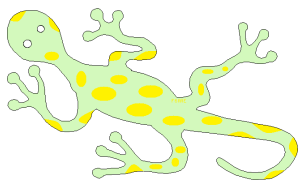
**Attention, ce programme est sans fin** et impose pour sa sortie une coupure alimentation hors de la fonction LECTURE sur l'HORLOGE.

Il ne faut que des "B" sur le plateau au lancement du programme sauf sous la tête de Lecture/Écriture où l'on doit positionner un "1".



Le temps d'exécution augmente régulièrement avec le nombre de pions à recopier de l'élément  $n$  sur le plateau à l'élément  $n+1$ .

Préférable au programme n°40.



### Programme utilisateur n°45.

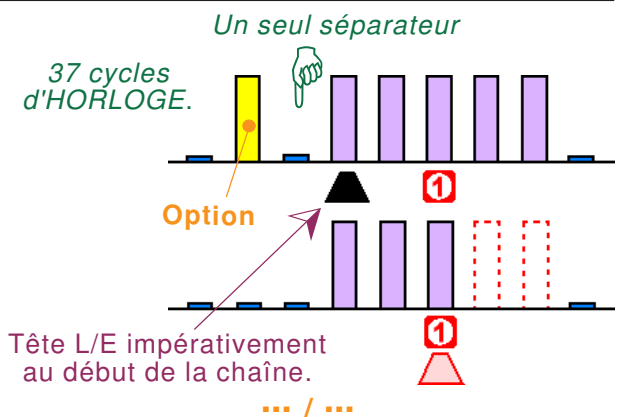
**R**éalise une **DICHOTOMIE** sur une chaîne **UNAIRE IMPAIRE**. En fonction du positionnement d'un BIT d'**Option** la zone effacée sera à gauche ou à droite de la donnée. Le centre de la chaîne **UNAIRE** sera positionné de préférence sur une origine de type ①.

**Option** = "0" effacement à Gauche.

**Option** = "1" effacement à Droite.

La chaîne **UNAIRE** doit comporter au moins trois BITS à "1" et la tête de L/E doit être initialisée à gauche de la donnée.

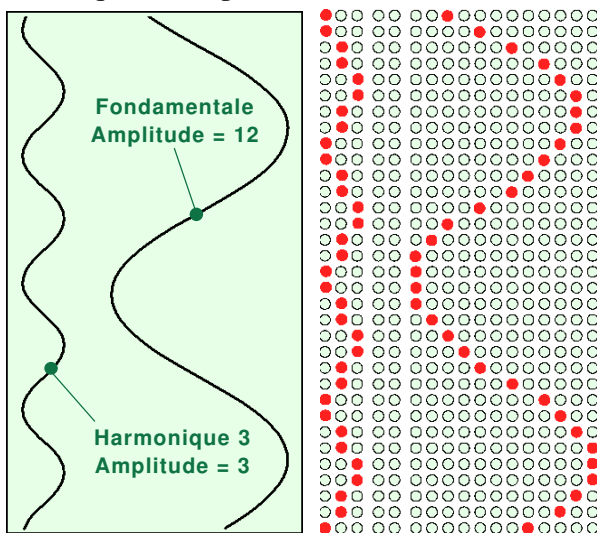
Il faut un seul séparateur entre le BIT d'**Option** et les éléments de la chaîne.



### Programme utilisateur n°46.

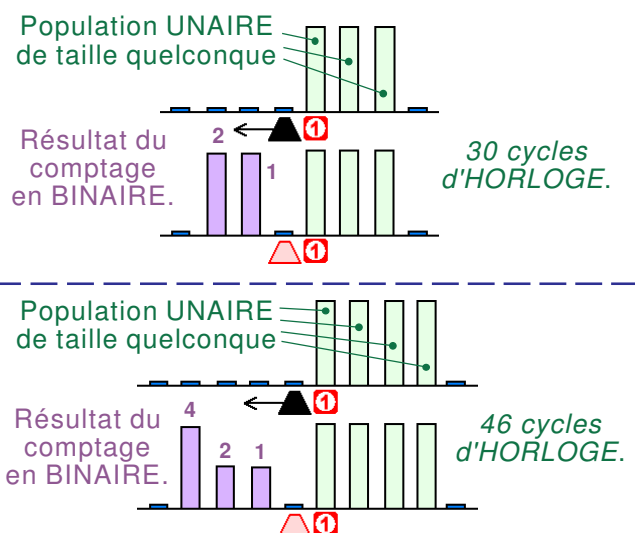
**C**ontrairement à un **Castor Affairé**, cet algorithme avec 66 instructions ne génère que quatre cycles d'HORLOGE pour se terminer. En outre, il inclut dix écritures surabondantes et 29 lignes de transitions inutiles non explorées par le programme.

**Cette fonction NON CALCULABLE** concrétise une "fantaisie artistique" qui utilise les trous de la feuille perforée pour créer le dessin géométrique de deux ondes sinusoïdales harmoniques d'amplitudes différentes.



### Programme utilisateur n°49.

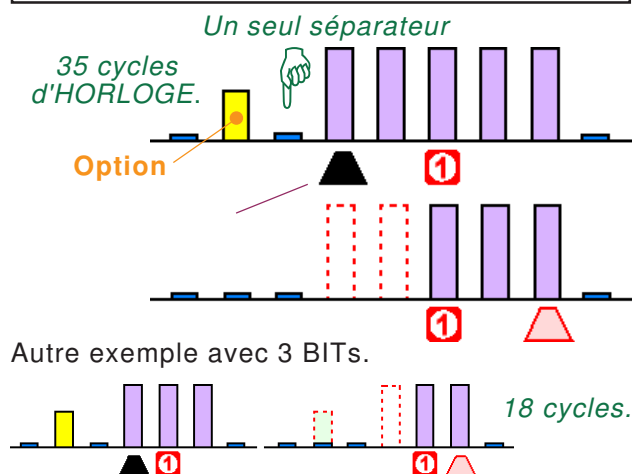
**R**épresentatif des premières tabulatrices électromécaniques utilisées aux USA pour les recensements ou le dépouillement des élections, ce programme simule un **comptage BINAIRE** pour recenser une population **UNAIRE**. Prévoir un nombre de "B" suffisants à gauche pour le compteur **BINAIRE**.



**ATTENTION :** Le nombre de cycles d'HORLOGE augmente très rapidement avec le nombre d'individus dans la population.

### Programme utilisateur n°45.

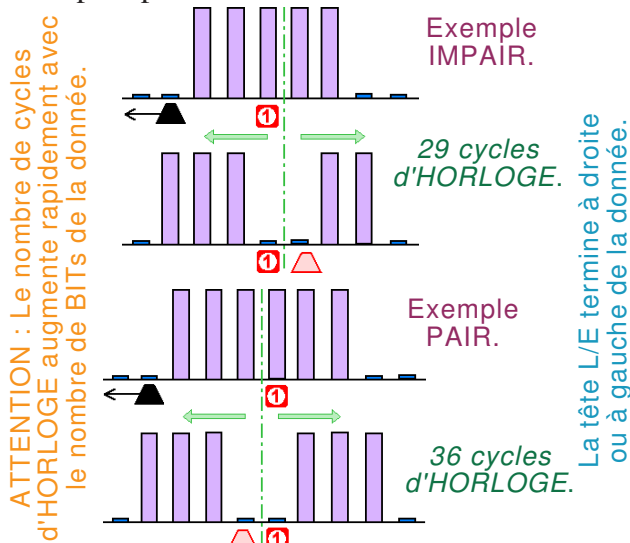
Efface le BIT d'**Option** en fin de traitement et laisse la tête de L/E sous l'origine ① si effacement à Droite, ou à Droite du résultat de la dichotomie si effacement à Gauche.



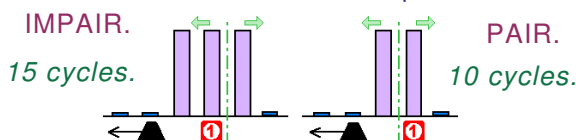
**NOTE :** En botanique, la dichotomie est la division d'un organe végétal en deux parties. En mathématique ou en informatique une dichotomie consiste à éliminer la moitié des éléments d'un ensemble quelconque "à sa droite" ou "à sa gauche".

### Programme utilisateur n°44.

**S**épare une chaîne de longueur quelconque en deux "moitiés" **UNAIRES**. Si le nombre de BITS de la donnée est **IMPAIR** la "moitié la plus grande" sera décalée à gauche et la plus petite à droite.



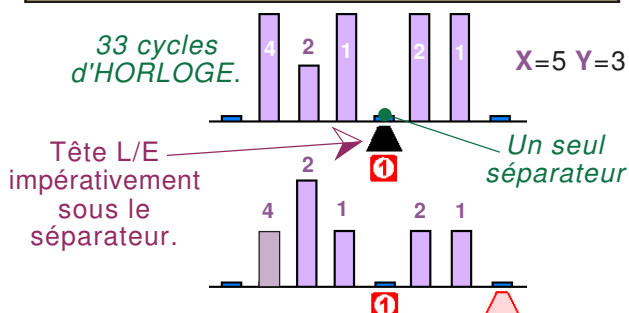
Deux autres exemples.



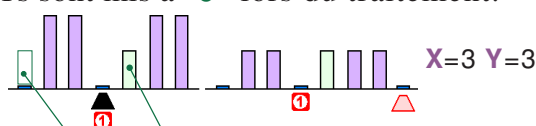
### Programme utilisateur n°48.

**S**oustraction de deux **BINAIRES**. Calcule la valeur de **X** en faisant  $X - Y$ . En fin de programme **Y** est forcée à zéro, les données étant toutes codées en **BINAIRE** pur.

- La tête de L/E doit impérativement se trouver sous la séparation de **X** et **Y**.
- Un seul "**B**" pour séparer **X** et **Y**.
- X** doit être supérieur ou égal à **Y**.



La valeurs de **Y** peut être nulle.  
La tête de L/E termine à droite de **Y** dont tous les BITS sont mis à "0" lors du traitement.

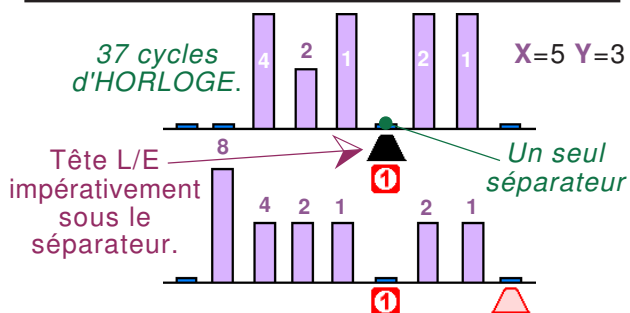


Des "0" en tête sur **X** et **Y** sont possibles.

### Programme utilisateur n°47.

**A**ddition de deux **BINAIRES**. Calcule la valeur de **X** en faisant la somme  $X + Y$ . En fin de programme **Y** est forcée à zéro, les données étant toutes codées en **BINAIRE** pur.

- La tête de L/E doit impérativement se trouver sous la séparation de **X** et **Y**.
- Un seul "**B**" pour séparer **X** et **Y**.
- Prévoir assez de "**B**" à gauche de **X**.



L'une des deux valeurs initiale peut être nulle.  
La tête de L/E termine à gauche de **Y** dont tous les BITS sont mis à "0" lors du traitement.



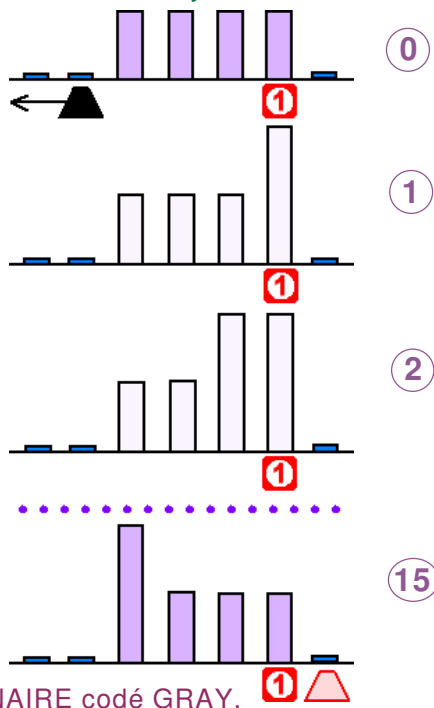
Des "0" en tête sur **X** et **Y** sont possibles.

### Programme utilisateur n° 50.

**R**éalise un **Compteur BINAIRE codé en GRAY**. Débute à la valeur zéro codée en GRAY comme en BINAIRE pur. Termine à la valeur quinze qui en code GRAY s'écrit "1000".

**ATTENTION : 240 cycles d'HORLOGE.**

Voir le dos de la Fiche n°52 pour le concept de déroulement de cet algorithme.



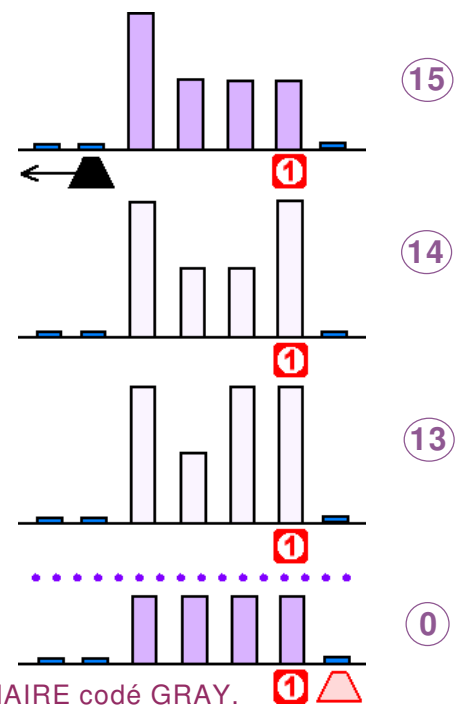
15 en BINAIRE codé GRAY.

### Programme utilisateur n° 51.

**R**éalise un **Décompteur BINAIRE codé en GRAY**. Débute à la valeur 15 codée en GRAY qui s'écrit "1000". Termine à la valeur zéro qui s'écrit comme en BINAIRE pur.

**ATTENTION : 307 cycles d'HORLOGE.**

Voir le dos de la Fiche n°52 pour le concept de déroulement de cet algorithme.



0 en BINAIRE codé GRAY.

### Programme utilisateur n° 50 & 51.

**E**lémentaires à décrire, ces programmes imposent un très grand nombre de cycles d'HORLOGE car il doivent effectuer régulièrement des déplacements de la tête de L/E à droite et à gauche de la donnée ce qui impose une foultitude de fonctions lecture.

#### ➤ Algorithme pour le COMPTEUR.

La donnée de départ est initialisée à "0000".

- Si le nombre actuel de "1" est PAIR alors inverser le BIT de poids faible à Droite.
- Si le nombre actuel de "1" est IMPAIR alors inverser le BIT situé à Gauche du BIT "1" se trouvant le plus à Droite de la donnée.
- Si la valeur actuelle vaut "1000" : **FIN**.

#### ➤ Algorithme du DÉCOMPTEUR.

La donnée de départ est initialisée à "1000".

- Si le nombre actuel de "1" est IMPAIR alors inverser le BIT de poids faible à Droite.
- Si le nombre actuel de "1" est PAIR alors inverser le BIT situé à Gauche du "1" se trouvant le plus à Droite de la donnée.
- Si la valeur actuelle vaut "0000" : **FIN**.

### Programme utilisateur n° 53.

**P**our déterminer le **PGCD** des deux nombres **X** et **Y** le programme utilise l'**algorithme d'Euclide** qui se résume à :

- Si  $X < Y$  alors enlever  $X$  à  $Y$ .
- Si  $X > Y$  alors enlever  $Y$  à  $X$ .
- Recommencer tant que l'un des deux n'est pas égal à zéro. Si c'est le cas  $X = \text{PGCD}$ .

#### Déroulement de l'algorithme.

- On force à "0" le "1" situé à Droite de **X**. On teste s'il reste encore un "1" à Gauche. Si ce n'est pas le cas on saute en (D). Si c'est le cas on saute en (B).
- On force à "0" le "1" situé à Gauche de **Y**. On teste s'il reste encore un "1" à Droite. Si ce n'est pas le cas  $Y = 0$  on va en (C). Si c'est le cas on recommence en (A).
- On remet tous les BITS de "**Y**" à "1", puis On revient sur **X**, on efface **Y** BITS. On teste s'il reste encore un "1" à Gauche. Si ce n'est pas le cas on saute en (E). Si c'est le cas on revient en (A).
- Traitement comme en (E) mais les rôles de **X** et de **Y** sont inversés et **G** devient **D**.
- La tête de L/E va à Droite de **X** et **FIN**.

### Programme utilisateur n°51.

N	Binaire pur	Code GRAI
15	1111	1000
14	1110	1001
13	1101	1011
12	1100	1010
11	1011	1110
10	1010	1111
09	1001	1101
08	1000	1100
07	0111	0100
06	0110	0101
05	0101	0111
04	0100	0110
03	0011	0010
02	0010	0011
01	0001	0001
00	0000	0000

### Programme utilisateur n°50.

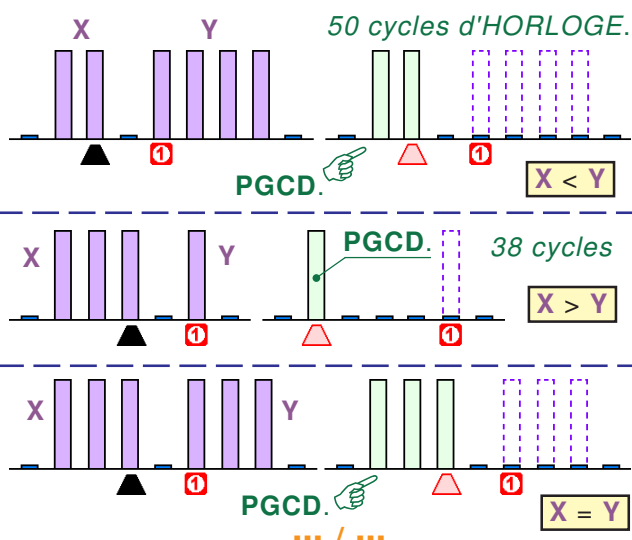
N	Binaire pur	Code GRAI
00	0000	0000
01	0001	0001
02	0010	0011
03	0011	0010
04	0100	0110
05	0101	0111
06	0110	0101
07	0111	0100
08	1000	1100
09	1001	1101
10	1010	1111
11	1011	1110
12	1100	1010
13	1101	1011
14	1110	1001
15	1111	1000

### Programme utilisateur n°53

**D**éterminer le PGCD de deux **UNAIRE**s **X** et **Y** quelconques. La donnée **X** est remplacée par le résultat également codé en **UNAIRE**. La donnée **Y** est effacée par des "B". Programme achevé la tête de L/E se trouve à droite du résultat. *En début de programme la tête de L/E doit se trouver à droite de X.*

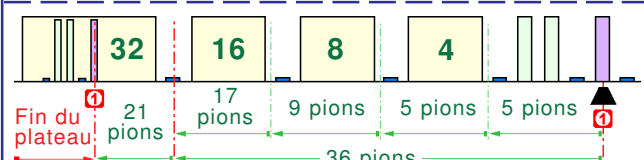
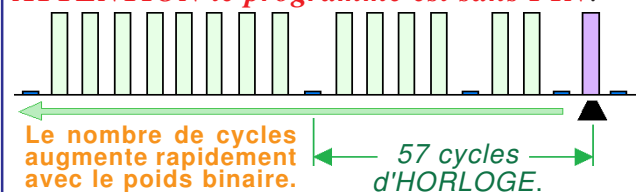
**X** peut être inférieur, égal ou supérieur à **Y**.

Le nombre de cycles d'HORLOGE augmente rapidement avec la taille de **X** ou (et) de **Y**.



### Programme utilisateur n°52.

**C**onstruit la suite des puissances de deux en **UNAIRE**. L'ensemble du barillet ne doit contenir que des "B" sauf sous la tête de Lecture/Écriture où doit se trouver un "1". **ATTENTION le programme est sans FIN.**



À partir de l'élément à 16 BITS, l'écriture de l'élément à 32 BITS sature le carrousel et l'exécution de l'algorithme n'est plus correcte.

#### Principe de l'algorithme :

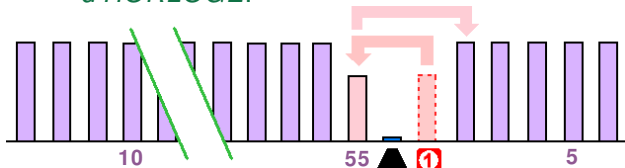
On duplique **n** en **n+1** à gauche en forçant les BITS de **n** à "0". Quand tous les BITS de **n** sont à "0" on recopie une deuxième fois des "1" à gauche de **n+1** en restituant les BITS de **n** à "1". Lorsque tous les bits de **n** sont à "1" on revient sur **n+1** qui se change en **n** et on recommence.



### Programme utilisateur n°54.

**P**etit programme élémentaire assez voisin d'un Castor affaîré. Le défi consiste à créer un programme avec le minimum d'instructions qui engendre en fonction de la donnée initiale proposée un maximum de cycles d'HORLOGE sur la machine. Transforme une suite composée d'un nombre quelconque de "1" en une suite de taille identique remplacée par des "0". **La taille la plus grande sur la machine sera de 55 BITS** placés à "1". **Cet algorithme est utilisé comme exemple didactique dans le tutoriel sur la Machine de Turing électronique à base d'une carte ARDUINO NANO.**

1598 cycles d'HORLOGE.

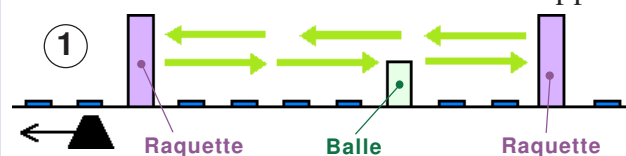


Le programme va remplacer tous les "1" par des "0" mais en partant alternativement des extrémités vers le centre obligeant ainsi à balayer "factoriellement" la zone des données dans un chassé/croisé de type :

1-55 / 55-2 / 2-54 / 54-3 / 3-53 / 53-4 / 4-52 ...

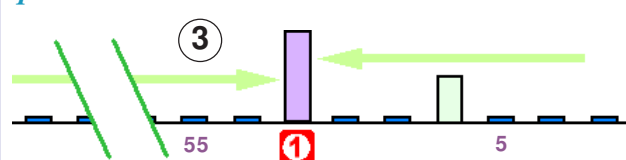
### Programme utilisateur n°57.

**S**imule symboliquement une partie de Ping-pong entre deux bornes à "1" qui représentent les joueurs. La distance minimale entre deux bornes est de deux pions. (Voir ②) La balle est simulée par un "0". L'espace maximal peut faire 55 BITS, comme en ③ et c'est un cas particulier où il n'y a qu'une seule borne à "1" la balle venant la "frapper"



alternativement à droite et à gauche. La balle peut être placée n'importe où entre les deux limites, y compris contre l'une d'entre elles.

**NOTE :** cet algorithme fonctionne parfaitement si on inverse tous les "G" et "D" et que la tête de L/E est à droite des données.



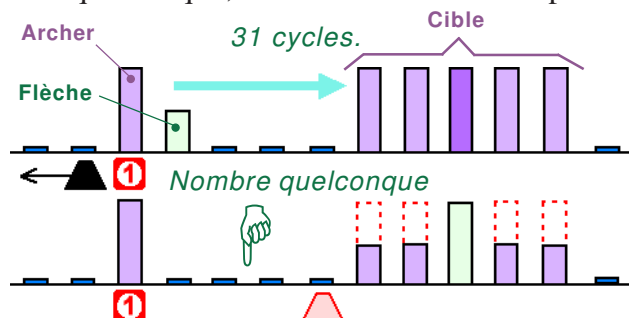
### Programme utilisateur n°58.

**A**lgorithme qui simule le tir d'une flèche en tir à l'arc au centre d'une cible. La cible est symbolisée par une suite de "0" en nombre IMPAIR. La flèche plantée au centre de la cible conservera l'état "1" et dépassera en hauteur.

**ATTENTION :** Le nombre de "1" pour la cible doit être IMPAIR.



Le nombre de "B" entre l'archer et la cible peut être quelconque, mais au moins trois espaces.

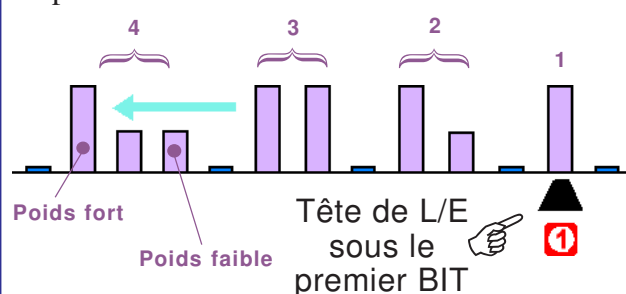


Le nombre de cycles d'horloge augmente avec la grandeur de la cible. Exemples :

10 Espaces / Cible 21 pions : 283 cycles.  
10 Espaces / Cible 41 pions : 943 cycles.

### Programme utilisateur n°61.

**C**onstruire la suite des Entiers codés en BINAIRE pur. **Noter que ce programme n'a pas de Fin.** La configuration de départ débute avec la première valeur initialisée manuellement à "1". La tête de L/E doit impérativement être sous ce BIT.



**NOTE :** L'algorithme a besoin de place à gauche des résultats pour construire la valeur suivante. Cette exigence, à partir de la valeur 12 incluse, engendre un déroulement du programme incorrect par écrasement des BITS des premières valeurs. Il devient pertinent de sortir de l'Exécution à partir du cycle d'HORLOGE n°414 soit 25 minutes de fonctionnement sur la machine.

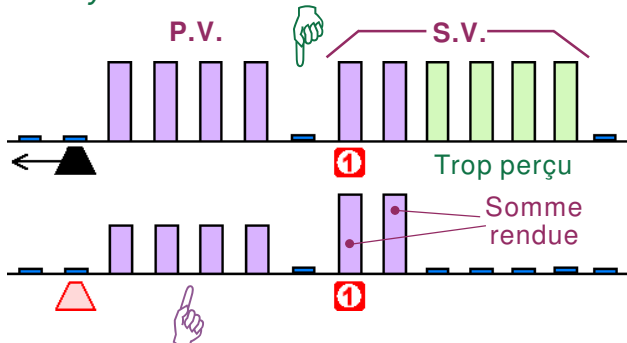
1100-1011-1010-1001-1000-111-110-101-100-11-10-1

### Programme utilisateur n°56.

**C**alcule la monnaie à rendre lors d'une transaction commerciale. Le prix de vente **P.V.** et la somme versée par l'acheteur **S.V.** sont codés en UNAIRE.

La somme versée **S.V.** sera impérativement supérieure ou égale à **P.V.** le prix de vente.

81 cycles Un seul séparateur



L'encaissement est simulé en forçant à "0" les BITS relatifs au **P.V.** convenu avec l'acheteur.

**NOTE :** Cet algorithme est assez boulimique en nombre de cycles d'HORLOGE. Exemples :

P.V.	S.V.	Nb cycles
3	4	50
5	5	90
4	33	297

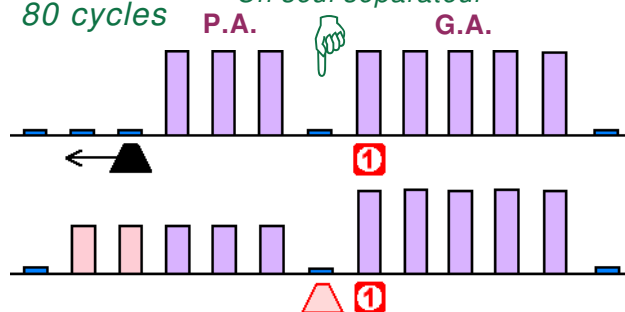
### Programme utilisateur n°55.

**S**imule le programme qui sur Apollo gérait la circularisation de l'orbite du train spatial.

Pour des raisons de stabilité du séjour du module resté en **orbite basse**, c'est le **petit axe de l'ellipse** qui était augmenté au périlune et était égalisé au grand axe.



80 cycles Un seul séparateur



- **P.A.** : Petit Axe. (Il faut **P.A.** > 1)
- **G.A.** : Grand Axe. (Il faut **G.A.** > **P.A.**)
- La tête de L/E doit être à gauche de la donnée représentant **P.A.**

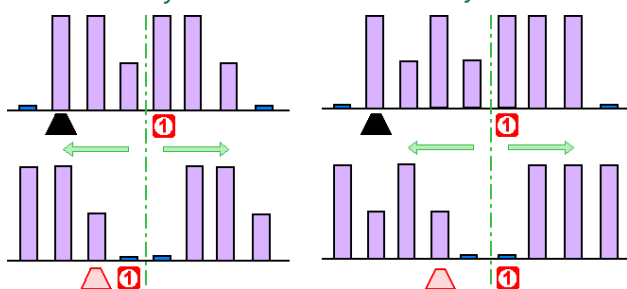
Si **P.A.** = 20 et **G.A.** = 25 il faut 1375 cycles HORLOGE et plus d'une heure machine.

### Programme utilisateur n°60.

**A**lgorithme qui sépare une donnée BINAIRE quelconque en deux moitiés de tailles analogues séparées par deux "B". Si le nombre de BITS et PAIR les deux entités seront de même taille. Si la chaîne est de taille IMPAIRE la donnée la plus "longue" sera celle de gauche. La tête de L/E sera à droite du morceau situé à gauche après le traitement.

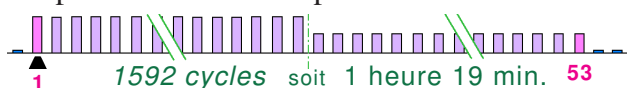
41 cycles.

51 cycles.



**ATTENTION :** La tête de L/E doit se trouver sur le pion de gauche de la donnée BINAIRE.

La donnée de taille la plus grande sera de 53 BITS pour laisser deux "B" pour l'écart à la coupure et un "B" de séparation des extrémités.



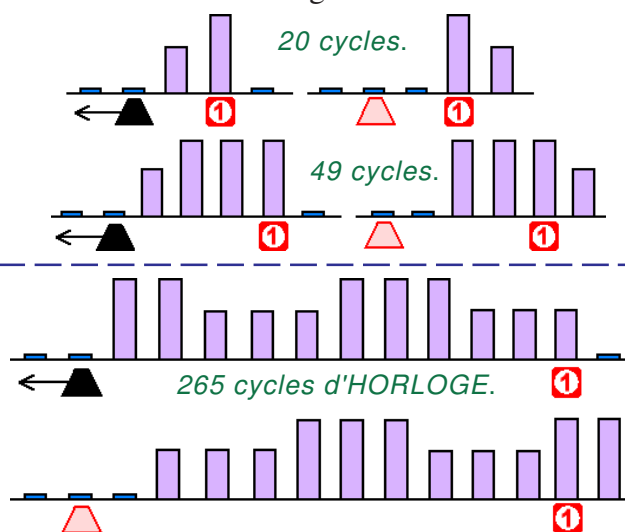
### Programme utilisateur n°59.

**I**nverser latéralement une donnée binaire.

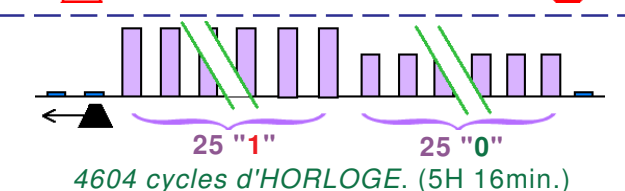
(La retourner horizontalement.) Par rapport à sa position initiale, la donnée est décalée d'un BIT à droite. La tête de L/E se trouvera à deux "B" à gauche du résultat final.

20 cycles.

49 cycles.



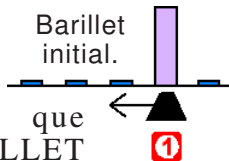
265 cycles d'HORLOGE.



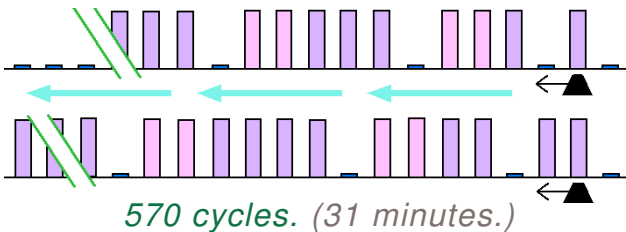
25 "1" 25 "0" 4604 cycles d'HORLOGE. (5H 16min.)

### Programme utilisateur n°62.

**C**onstruire sous forme **UNAIRE** la suite  $U^{n+1} = U^n + 2$ . Globalement, la technique consiste à recopier la valeur de  $U^n$  à gauche de cette dernière et à y concaténer deux BITS "1" à sa gauche. Le nombre de cycles d'Horloge ainsi que l'encombrement du BARILLET augmentent vite avec la taille de la donnée.



**13-11-9-7-5-3-1** : Sature le BARILLET à **13**.  
783 cycles. (43 minutes.)



**10-8-6-4-2** : Sature le BARILLET à **10**.

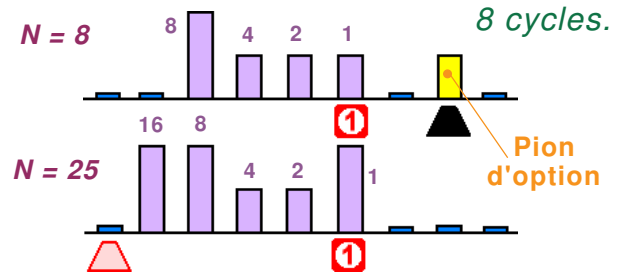
Exemples "typiques" :

Nb BITS	Nb cycles	Saturation à	Tmp Machine
10	1084	16	56 minutes
20	907	22	46 minutes
25	1380	27	1H 9 minutes

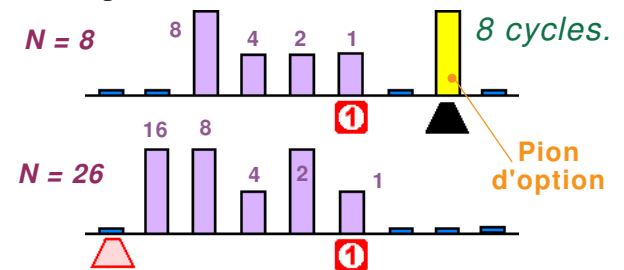
### Programme utilisateur n°65.

**F**onction d'un **BIT d'option** calcule les valeurs de  $3 * N + 1$  ou de  $3 * N + 2$  codées en **BINAIRE pur**. Peut calculer sur des valeurs **BINAIRE**s vraiment très élevées et traite très rapidement. (52 BITS.) **Le BIT d'option doit se trouver à droite de la donnée BINAIRE et avec un seul "B" de séparation.**

BIT d'Option = "0" :  $3 * N + 1$ .

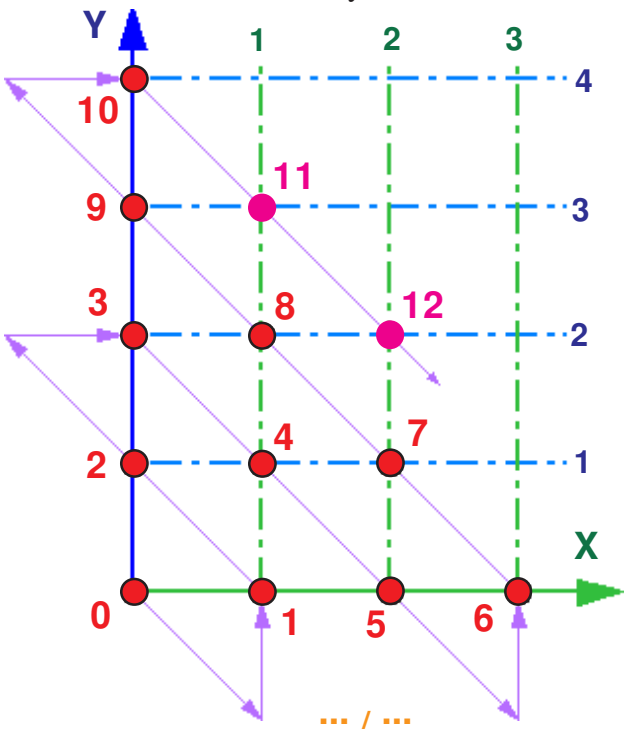


BIT d'Option = "1" :  $3 * N + 2$ .



### Programme utilisateur n°66.

**E**tablir une **Bijection** entre les entiers naturels **N** et les points du plan de coordonnées **X, Y entières et positives**. Cet algorithme ne fonctionne que jusqu'à la valeur 10 obtenue à 57 cycles d'HORLOGE.



### Programme utilisateur n°66.

N	Points du plan X,Y	En Binaire X,Y	Sur la Machine X,Y	Nb Cycles
0	(0,0)	(0,0)	(0,0)	0
1	(1,0)	(1,0)	(1,0)	5
2	(0,1)	(0,1)	(0,1)	12
3	(0,2)	(0,10)	(0,01)	17
4	(1,1)	(1,1)	(1,10)	22
5	(2,0)	(10,0)	(10,0)	29
6	(3,0)	(11,0)	(11,0)	31
7	(2,1)	(10,1)	(10,10)	41
8	(1,2)	(1,10)	(01,01)	48
9	(0,3)	(0,11)	(00,11)	53
10	(0,4)	(0,100)	(00,001)	57
11	(1,3)	(1,11)	(01,110)	0
12	(2,2)	(10,10)	(10,010)	0
13	(3,1)	(11,1)	(11,100)	0
14	(4,0)	(100,0)	(100,000)	0
15	(5,0)	(101,0)	(101,000)	0
16	(4,1)	(100,1)	(100,100)	0

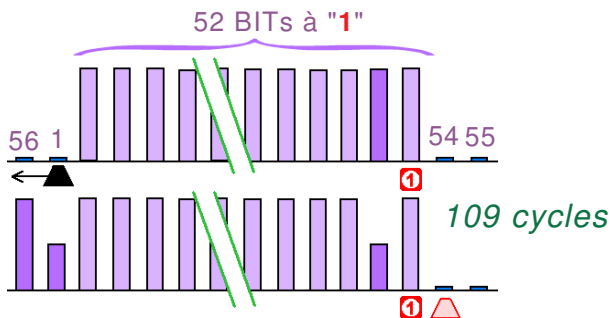
↔ et ↔ : Sens de lecture de **X** et de **Y**.

### Programme utilisateur n°64.

**L** *algorithm* calcule  $3 * N$  en **BINAIRE pur**. Il peut traiter des valeurs vraiment très grandes. Maximum 2 élevé à la puissance 52 soit la bagatelle de 9.007.199.254.740.991 !



**Le traitement est très rapide.** Pour exemple on va prendre la valeur 2 élevé à la puissance 52 :

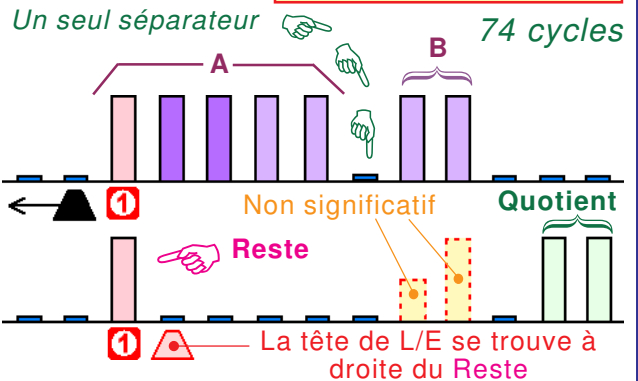


Comme le calcul ne consomme que quatre transitions et que les rotations du plateau sont rapides, on dégage de deux pions la tête de L/E à gauche du résultat.

### Programme utilisateur n°63.

**D** *ivision euclidienne* de l'entier **A** par l'entier **B**. Les deux entiers **A** et **B** sont codés en **UNAIRE**.

Avec **A** > ou égal à **B**.



A	B	Reste	Quotient	Nb Cycles
1	1	0	1	13
13	4	1	3	312
27	7	6	3	1095
25	15	10	1	1115
30	19	11	1	1578

**NOTE :** Si on initialise le barillet avec **A** < **B**, la tête de L/E termine sous le séparateur des deux données et **A** "0" sont à gauche de **B**.

Exemple : **A** = 4 et **B** = 8 : 1111-00001111

### Programme utilisateur n°66.

Evolution des données durant la phase où l'algorithme fonctionne correctement jusqu'au cycle d'horloge n°57. La position de la tête de L/E est repérée en couleur orange sur les "0", les "1" ou les séparateurs représentés par "·".

00	0-0---	0	18	0-01---	36	11-10---
01	1-0---		19	1-01---	37	11-00---
02	1-0---		20	1-01---	38	11-00---
03	1-1---		21	1-11---	39	11-10---
04	1-1---		22	1-10---	40	1-1-10---
05	1-0---	1	23	1-10---	41	10-10---
06	1-0---		24	1-10---	42	10-10---
07	1-1---		25	0-10---	43	10-00---
08	1-1---		26	10-10---	44	10-01---
09	0-1---		27	10-10---	45	10-01---
10	0-1---		28	10-10---	46	10-01---
11	0-0---		29	10-00---	47	1-1-01---
12	0-01---	2	30	10-00---	48	0-1-01---
13	0-01---		31	1-1-00---	49	0-1-01---
14	0-01---		32	1-1-00---	50	0-1-01---
15	1-01---		33	1-1-10---	51	0-1-1-1---
16	1-01---		34	1-1-1-1---	52	0-1-1-1---
17	0-01---	3	35	1-1-1-1---	53	0-0-1-1---
						9

### Programme utilisateur n°66.

La machine démarre sur [0,0] **X** et **Y** étant séparés par un "B". et exprimés en **BINAIRE pur**. Au début la tête de lecture est sous **X**.

**X** se lit de la Droite vers la Gauche.

**Y** se lit de la Gauche vers la droite.

