

Ce document appartient à :

Nom:

Prénom :

Année :

Sciences et Techniques Industrielles « STI »

Automatismes industriels

Cours

Imprimées sur papier blanc
à acquérir et connaître sans support

- ✓ i2241 : Mise en œuvre des détecteurs
- ✓ i2441 : Mise en œuvre des capteurs analogiques
- ✓ i2511 : Algorithmique CEI 61131-3 – Programmation Ladder
- ✓ i2521 : Calcul numérique sur automates programmables – Codage binaire
- ✓ i2571 : Grafcet

**Section de Technicien Supérieur en Électrotechnique
Étudiants et Apprentis**

Mise en œuvre des détecteurs

1. Détection par contact

1.1. Détecteur électromécanique, sortie à contacts secs



Description

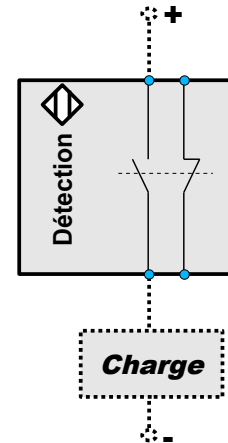
Ce type de détecteur comporte généralement 2 contacts électriques (un NO et un NF). Ils peuvent être utilisés sous différentes tensions (inférieure à la tension maximum admissible) et ne sont pas polarisés.

Ce type de détecteur est souple d'utilisation mais subit l'usure des contacts électriques.

Il est néanmoins relativement fiable.

Connexion

Il se branche comme un interrupteur, en série avec le circuit à contrôler.



2. Détection électronique

Ces détecteurs comportent un circuit électronique qui commande une ou plusieurs sorties statiques.

Ils sont pratiquement inusables car ils ne comportent pas de contacts électriques mobiles.

2.1. Sortie statique à 2 fils



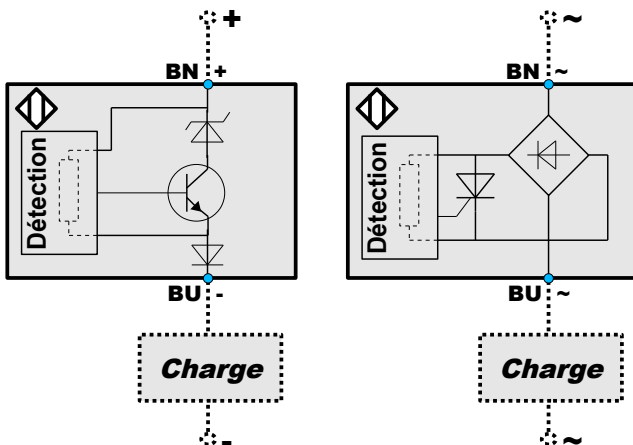
Description

Malgré la nécessité d'alimenter en énergie le circuit électronique, ce type de détecteur ne comporte que deux fils. Ils peuvent être alimentés selon les modèles sous tension continue, alternative, ou indifféremment.

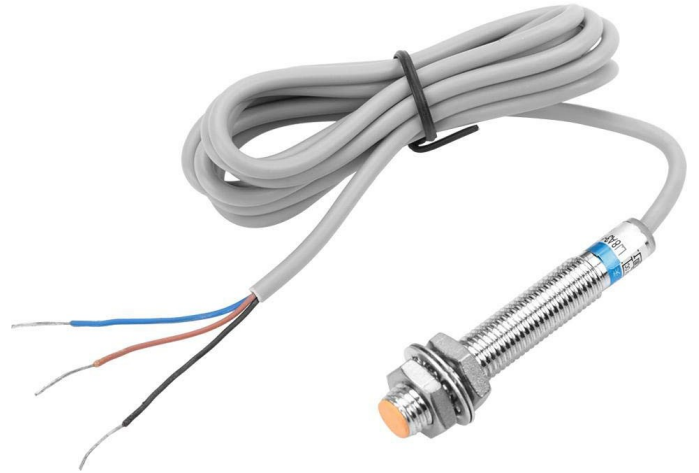
Il est utilisé lorsqu'il n'y a **pas nécessité d'une grande fréquence de commutation**. Dans le cas contraire, on préférera un détecteur 3 fils.

Connexion

Il est souple d'utilisation puisqu'il se connecte comme un détecteur à contacts secs, en série avec le circuit à contrôler.



2.2. Sortie statique à 3 fils



Description

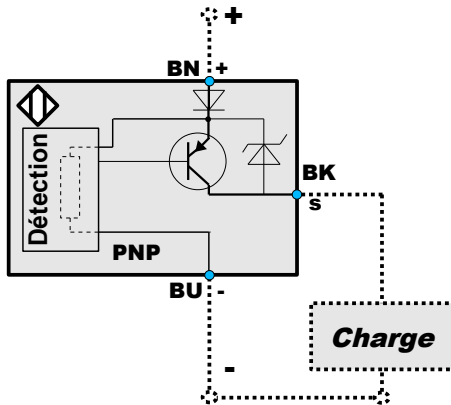
S'il ne comporte qu'une seule sortie statique, c'est un détecteur 3 fils sinon ça sera un 4 fils (2 sorties statiques).

Il fonctionne uniquement en tension continue.

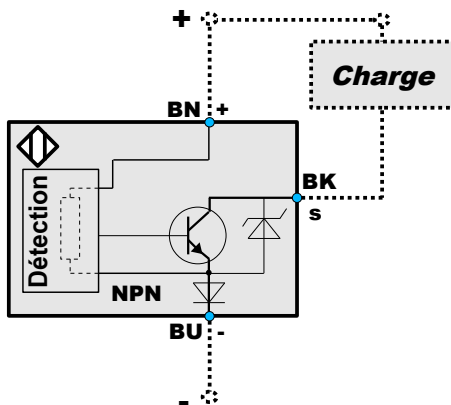
Il est utilisé lorsqu'il y a **nécessité d'une grande fréquence de commutation**. Dans le cas contraire, on pourra utiliser un détecteur 2 fils.

Connexion

Il peut être détecteur PNP ou NPN. Le détecteur PNP ou NPN comporte un transistor.

Pour le détecteur PNP :

Lorsque qu'il y a détection, le transistor est passant (contact fermé). Il va donc imposer le potentiel + sur la sortie S . La charge est branchée entre la sortie S et le potentiel -. Ce type de détecteur est adapté aux unités de traitement qui fonctionnent en **logique positive**.

Pour le détecteur NPN :

Lorsque qu'il y a détection, le transistor est passant (contact fermé). Il va donc imposer le potentiel - sur la sortie S. La charge est branchée entre la sortie S et le potentiel + . Ce type de détecteur est adapté aux unités de traitement qui fonctionnent en **logique négative**.

On prendra donc soin d'identifier le type de logique utilisée par les unités de traitement (Automate programmable, etc...).

2.3. Sortie statique, 4 fils

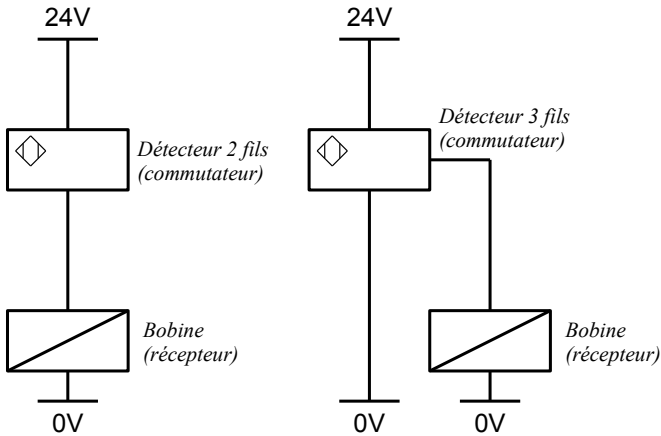
Un détecteur 4 fils est un détecteur 3 fils comportant un fil supplémentaire pour la transmission d'un signal complémentaire.

On dispose donc d'une sortie passante en présence d'une pièce (**Normalement ouvert**, « NO »), l'autre bloquée en présence d'une pièce (**Normalement fermée** « NC »).

3. Mise en œuvre

3.1. Schéma

Un cadre contenant le symbole de détecteur électronique



3.2. Repérage

Les différents fils sont souvent repérés par leur couleur, notée en abrégé :

Abrégé	Complet	Couleur
BN	Brown	Marron
BK	Black	Noir
BU	Blue	Bleu

3.3. Compatibilité

Pour toute connexion d'un générateur à un récepteur, il faut vérifier les compatibilités en tension et en courant.

Dans tous les cas, il convient de tenir compte :

- ❖ **du courant commuté I_{max}** que peut fournir le détecteur lorsque celui-ci est passant. On vérifiera qu'il est supérieur au courant I_{IH} imposé par la charge :

$$I_{MAX} \geq I_{IH}$$

- ❖ **de la tension d'alimentation ou de service :** vérifier que la tension utilisée est bien dans les limites supportées par le détecteur.

Détecteurs 2 fils

Dans le cas des détecteurs 2 fils, il convient de tenir compte :

- ❖ **du courant résiduel I_R :** courant traversant le détecteur à l'état bloqué. Il doit être suffisamment petit pour ne pas rendre actif l'organe piloté par le détecteur (bobine, entrée d'API, ...).
- ❖ **de la tension de déchet U_D :** tension aux bornes du détecteur à l'état passant (mesurée pour le courant nominal). Elle doit être suffisamment petite pour ne pas bloquer l'entrée pilotée. On vérifiera :

$$V_{CC} - U_D \geq V_{IHmin}$$

Détecteurs 3 fils

Dans le cas des détecteurs 3 fils, il convient de tenir compte :

- ❖ **de la tension de saturation V_{SAT} :** c'est la tension entre la sortie du détecteur (collecteur) et l'entrée d'alimentation qui n'est pas reliée à la charge (émetteur) = tension collecteur-émetteur du transistor interne. Elle est parfois appelée aussi tension de déchet dans certains catalogues. On vérifiera que la tension restante de l'alimentation moins le déchet est suffisante pour la charge :

$$V_{CC} - V_{SAT} \geq V_{IHmin}$$

1. Rappel de la fonction de base



Mesurer, transformer l'amplitude d'une grandeur physique en une grandeur électrique exploitable dans un automatisme.

2. Signaux de sortie

Il s'agit d'être vigilant quant aux termes employés pour désigner une information de sortie :

- ◆ sortie **de** vitesse : information portée par le signal
- ◆ sortie **en** courant : grandeur électrique qui porte le signal

Attention alors au sens de :

- ◆ sortie **de** courant **en** tension :
= mesure d'un courant de 0 à 100A
sous la forme d'une tension de 0 à 5V

2.1. Sortie en tension

La tension de sortie est fonction linéaire de la gamme de mesure. Exemple : la tension varie de 0 à 10 V lorsque la hauteur mesurée varie de 0 à 5 mètres. Le calibre est souvent désigné par ses valeurs mini et maxi séparées par un tiret ; exemple : 0-10V

➤ Formats rencontrés :

- ◆ **0-10 V** : le plus répandu
- ◆ **± 10 V** : 2 sens de marche : rotation, ...
- ◆ 0-5 V : pour des cartes moins industrielles
- ◆ 1-5 V : pour adaptation avec 4-20mA (voir ce calibre)
- ◆ ...

➤ Avantages :

- ◆ Très facile à mettre en œuvre
- ◆ Impédance élevée (100 K Ω à plusieurs M Ω) pour entrée en tension : un simple potentiomètre peut être utilisé.

➤ Inconvénients :

- ◆ Risque d'atténuation et de déformation du signal sur de longues distances

2.2. Atténuation d'un signal

Voir en bas de page

2.3. Sortie en courant

Le courant de sortie est fonction linéaire de la gamme de mesure. Exemple : variation de 0 à 20 mA pour une mesure de 0 à 100°C.

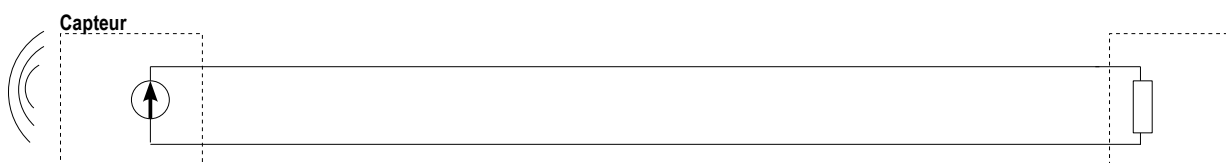
➤ Formats rencontrés :

- ◆ 0-20 mA : fréquent, non autonome en énergie
- ◆ **4-20 mA** : le plus répandu ; permet de détecter les ruptures de conducteur ou les défauts de mesure (Si I < 4mA)
- ◆ 0-10mA, 0-16mA : rarement

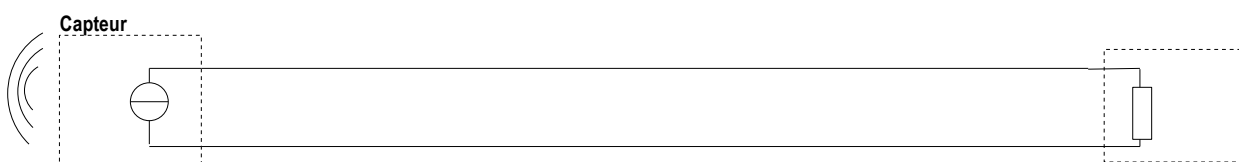
➤ Avantages :

- ◆ **Pas de perte de signal** sur de longues distances : le courant est le même dans toute la boucle.
- ◆ Peu sensible aux parasites électromagnétiques
- ◆ détection de rupture du conducteur de signal

➤ Sortie en tension



➤ Sortie en courant



2.4. Sortie bas-niveau

Certains capteurs ne disposent pas d'amplificateur de signal en sortie, et ne proposent donc pas de signal « haut niveau ».

Le signal « bas niveau », non normalisé, peut être de la forme :

- ◆ 0-50 μV , 0-200 μV , 0-1 mV, 0-63 mV, 0-100 mV, ...

2.5. Sortie en fréquence

La fréquence du signal de sortie est fonction de la grandeur à mesurer : débitmètre à effet hall, ...

L'amplitude peut varier en même temps mais la linéarité en amplitude est médiocre.

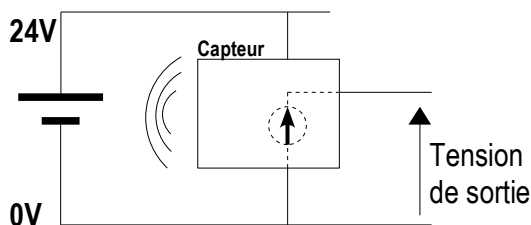
On emploiera le plus souvent un adaptateur fréquence/ tension pour utiliser le signal analogique sur un API.

3. Mise en œuvre

3.1. Alimentation, charge

3.1.1. Sortie en tension

Les capteurs à sortie **en** tension nécessitent généralement une alimentation continue externe. Le signal de sortie, en tension, est fourni en référence au 0 de l'alimentation.

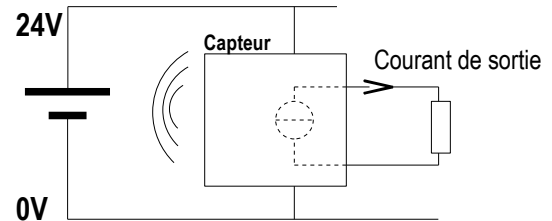


Toujours vérifier que la **résistance de charge n'est pas trop petite**, et que le courant qu'elle demandera pourra être fourni par le capteur.

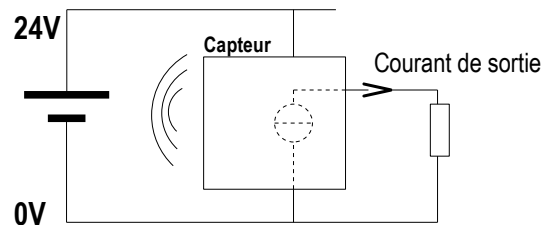
3.1.2. Sortie en courant

➤ Capteur 4 fils

Pour des sorties 0-20mA, ou des sorties 4-20mA de capteurs évolués.

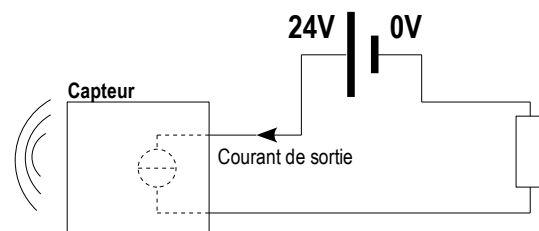


➤ Capteur 3 fils



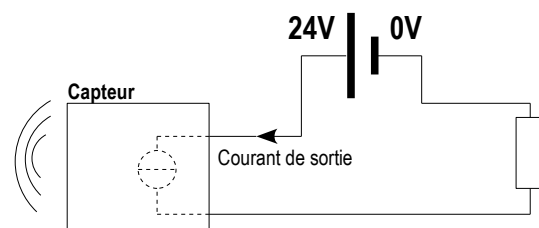
➤ Capteur 2 fils

Typiquement pour des sorties 4-20mA de capteurs peu évolués.



Vérifier que la **résistance de charge n'est pas trop grande**, sinon le capteur ne pourra pas délivrer une tension suffisante pour le courant qu'il veut fournir.

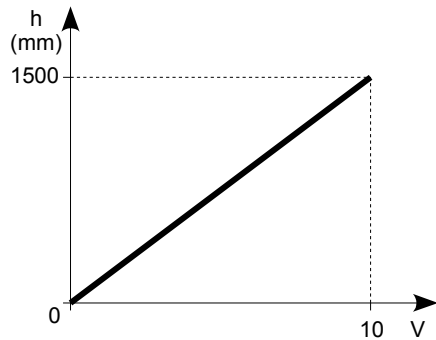
➤ Intégration d'un appareil de mesure



3.2. Interprétation de la mesure

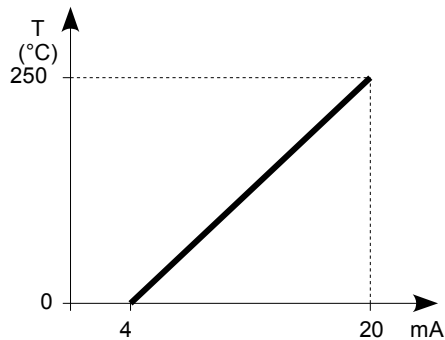
3.2.1. Sortie de niveau en tension

Détail des calculs :



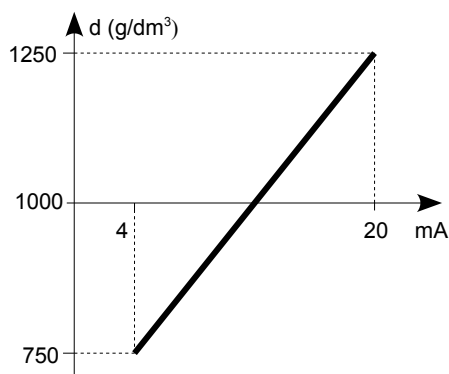
$$V_o = 6V \rightarrow h = ?$$

3.2.2. Sortie de température en courant



$$I_o = 13\text{mA} \rightarrow T = ?$$

3.2.3. Sortie de densité en courant



$$I_o = 7\text{mA} \rightarrow d = ?$$

Programmation des Automates selon CEI 61131-3

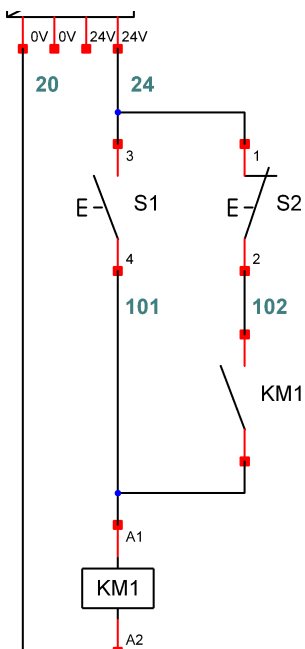
1. Les langages de programmation

La norme CEI 61131-3, intitulée « *Automates programmables - Partie 3 : Langages de programmation* », est une norme industrielle de la Commission électrotechnique internationale (CEI). Publiée la première fois en 1993, sa dernière édition date de septembre 2013.

Cette norme définit pour les automates programmables :

- ❖ 2 langages de programmation **textuels** :
 - liste d'instructions (**IL**, Instruction List),
 - texte structuré (**ST**, Structured Text),
- ❖ et 2 langages de programmation **graphiques** :
 - diagramme à contacts (**LD**, *Ladder Diagram*),
 - diagramme de bloc fonctionnel (**FBD**, Function Block Diagram),
 - ❖ un autre ensemble d'éléments graphiques et textuels appelé "diagramme fonctionnel séquentiel" (**SFC**, Sequential Function Chart) pour structurer l'organisation interne des programmes pour automate programmable et des blocs fonctionnels.
- Le SFC n'est donc pas un langage de programmation et ne doit pas être confondu avec le **GRAFCET** (GRAphe Fonctionnel de Commande Etape Transition), langage de spécification pour les diagrammes fonctionnels en séquence défini par la norme NF EN 60848.

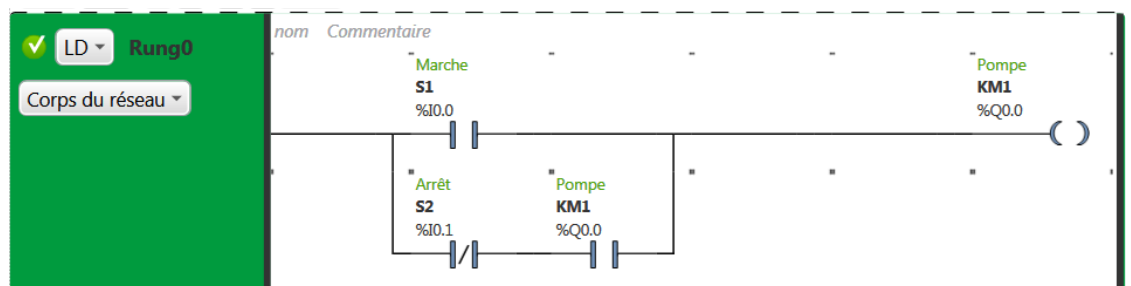
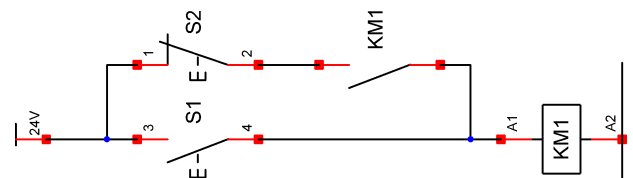
1.1. Langage LADDER



Le langage à contacts est une représentation des circuits électriques traditionnels. Il a été créé afin de faciliter la transition des électromécaniciens habitués à la logique câblée vers la logique programmée des API.

Pivot à gauche du schéma → :

Langage à contact ↓ :



2. Les conditions en LADDER



❖ Si un état est présent ...

❖ Détecter la présence d'une pièce...

❖ Détecter une pièce...



❖ Lorsqu'un événement survient...

❖ Détecter un événement à un instant précis...

❖ Lorsqu'un état apparaît...

❖ Lorsqu'une pièce arrive...

❖ Détecter l'arrivée d'une pièce...



❖ Lorsqu'un état disparaît...

❖ Lorsqu'une pièce part...



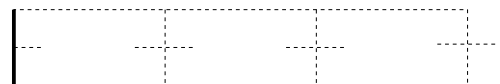
❖ Si une variable égale une certaine valeur...

Application : comptage



❖ Si une variable a atteint un certain seuil...

Application : analogique, contrôle de niveau, ...

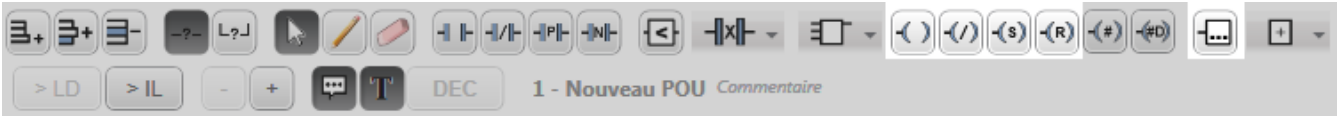


❖ Si une variable a dépassé un certain seuil...

Application : analogique, contrôle de niveau, alerte de débordement, ...



3. Les actions en LADDER



- ❖ Activer une sortie TOR.
- ❖ Piloter un actionneur.
- ❖ Envoyer un ordre.

Tant que la condition est présente, la sortie est active.



- ❖ Mettre à 1 et conserver l'état.



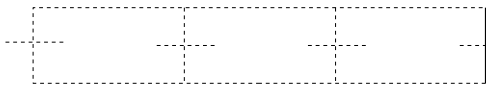
- ❖ Mettre à 0 et conserver l'état.



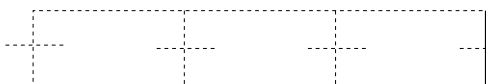
- ❖ Calculer l'opération [...].



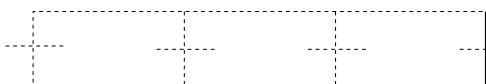
- ❖ Incrémenter (= ajouter 1)



- ❖ Affecter une valeur constante à une variable.



- ❖ Copier le contenu d'une variable.
- ❖ Stocker une valeur.
- ❖ Garder un mot identique pour l'utiliser à autre chose.



- ❖ Acquérir / Capturer / Bloquer la valeur d'une entrée analogique

Le calcul numérique sur les Automates Programmables

1. Arithmétique binaire

1.1. Association de caractères binaires

Un BIT (**B**INARY **digi**T) peut contenir 2 valeurs distinctes : 0 ou 1.

L'association d'un bit supplémentaire permet de distinguer 2 fois plus de valeurs :

En associant un 0 aux possibilités déjà présentes :

0 0 ou 0 1

En associant un 1 aux possibilités déjà présentes :

1 0 ou 1 1

En généralisant, dans une variable à n bits, on peut donc stocker 2^n valeurs différentes.

Exemple

Sur 3 bits, on peut stocker $2^3 = 8$ valeurs différentes :

000, 001, 010, 011, 100, 101, 110, 111

A retenir

Sur 4 bits, on peut coder16 valeurs différentes.

Sur 8 bits, on peut coder256 valeurs différentes.

Sur 16 bits, on peut coder 65536 valeurs différentes.

1.2. Identification à des valeurs décimales

Dans le cas général d'une base n à convertir en base 10, chaque caractère représenté sera associé à un *poids* selon son rang, du plus faible à droite au plus fort à gauche, d'une valeur $n \times$ plus grande que le précédent à droite.

Dans le cas plus précis du binaire, chaque bit représenté sera associé à un *poids binaire* selon son rang, du plus faible à droite au plus fort à gauche, d'une valeur $2 \times$ plus grande que le précédent à droite.

Représentation

Poids	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Valeur	128	64	32	16	8	4	2	1
N ^{bre} binaire	b7	b6	b5	b4	b3	b2	b1	b0

Exemple :

Valeur	128	64	32	16	8	4	2	1	
Bits	×	0	0	1	0	1	1	0	1
	=	0	0	32	0	+8	+4	0	+1
									= 45 ₍₁₀₎

Exercices

Convertir en base 10 :

• 100 1110 0010 1001₍₂₎

• 64223₍₇₎

• 3A8₍₁₃₎

• 4E29₍₁₆₎

1.3. Représentation hexadécimale

A partir de la représentation binaire, on peut reconstituer très facilement la représentation hexadécimale par le groupement par 4 bits à partir de la droite, suivi de la conversion de chaque quartet en caractère hexadécimal.

1.4. Conversion décimal / binaire

- Énumérer les poids (puissances de 2 vers la gauche) jusqu'à dépasser la valeur à convertir,
- Pour chacun des poids inférieurs,
 - si inférieur ou égal à la valeur convertie, le soustraire et écrire un 1
 - sinon écrire un 0.

Exercices

Convertir en binaire :

- 1968 (10)
- 2005 (10)
- 544 (10)
- 2015 (10)

1.5. Opérations arithmétiques

L'addition

Revenons aux bases de l'addition en posant les opérations :

$$\begin{array}{r}
 157 \\
 + 77 \\
 \hline
 234
 \end{array}
 \quad
 \begin{array}{r}
 1001 \\
 + \\
 \hline

 \end{array}$$

La multiplication

$$\begin{array}{r}
 57 \\
 \times 23 \\
 \hline
 171 \\
 + 114 \\
 \hline

 \end{array}
 \quad
 \begin{array}{r}
 \\
 + \\
 \hline

 \end{array}$$

2. Les types de données

2.1. Type « Entier non signé »

Tous les bits de la donnée représentent un poids binaire. La valeur résultante est donc forcément entière et positive.

Sur n bits, distinguant 2^n valeurs distinctes, on pourra alors coder les valeurs 0 à $2^n - 1$

Poids	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Nbre binaire	b7	b6	b5	b4	b3	b2	b1	b0

Exemple :

Entier non signé sur 8 bits :

$2^8 = 256$ valeurs positives possibles : 0 à 255

2.2. Type « Entier signé »

Il est parfois nécessaire de travailler sur des nombres négatifs. Il s'agit alors de distinguer le signe de la valeur (+ ou -) par un bit supplémentaire. La taille des mémoires des processeurs étant limitée, le bit le plus à gauche sera réquisitionné pour représenter le bit de signe :

0 : nombre positif, 1 : nombre négatif.

Poids	S	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Nbre binaire	b7	b6	b5	b4	b3	b2	b1	b0

On remarquera cependant que l'on peut représenter la valeur nulle par +0 ou -0, ce qui entraînerait des problèmes de calcul.

Les valeurs négatives seront alors représentées en mode « complément à 2 » sur une taille (nombre de bits) dépendant du processeur utilisé (8 bits, 16 bits, 32 bits, ...).

2.3. Le format « complément à 2 »

Il s'agit de représenter les valeurs négatives par les opérations suivantes, à partir de la valeur positive :

- Complément à 1 (inverser chaque bit)
- Additionner 1

Exemple :

en considérant que l'on travaille sur des formats 8 bits :

$45_{(10)}$:		^s	0	0	1	0	1	1	0	1
Complément à 1 :			1	1	0	1	0	0	1	0
Addition de 1 :	+									1
Résultat :			1	1	0	1	0	0	1	1

= - $45_{(10)}$

Astuce :

Autre méthode pour calculer l'opposé d'un nombre binaire (en complément à 2) :

Recopier les bits en partant de la droite jusqu'au premier 1 rencontré inclus, puis inverser tous les autres bits à sa gauche.

Exercices – Calculs sur formats 8 bits signés

- Calculer l'opposé de $0_{(10)}$ en format binaire complément à 2 par le calcul arithmétique et non par l'astuce.
- Calculer en binaire $(+45) + (-45)$, $(+60) + (-45)$, $(+45) + (-34)$
- Calculer en binaire (-126) , (-127) , en déduire ce que serait (-128) .
- Convertir un nombre négatif binaire en décimal : 10011011

2.4. Type « Réel »

Il est parfois nécessaire de travailler sur des nombres décimaux, « à virgule ». Mais il ne sera pas possible de représenter l'ensemble des nombres réels, leur nombre étant infini, contrairement aux capacités d'un processeur qui sont limitées et finies.

Il s'agira alors de définir des limites aux parties entières (plus grande valeur absolue, avant la virgule) et aux parties décimales (limite de précision, après la virgule). La combinaison de ces deux contraintes est définie par le « **nombre de chiffres significatifs** », la virgule pouvant être placée n'importe où. On parle alors de **nombres « à virgule flottante »** ou « **flottants** ».

Le format plus complexe (**mantisse + exposant**) est présenté dans la documentation associée à ce cours. Le nombre de chiffres significatifs (précision) est lié à la taille mémoire occupée par la valeur : 4 octets, 8 octets, 10 octets, ...

Ce type de données n'est pas disponible sur les API d'entrée de gamme car leur puissance de calcul n'est pas suffisante.

2.5. Type « Caractère »

La représentation des caractères n'est qu'une interprétation de l'utilisateur. Chaque caractère est associé à une valeur numérique binaire selon un codage normalisé, et son affichage est assuré par une description de table de caractère, que l'homme sait reconnaître.

Les codages standards sont :

- Le format ASCII (depuis les années 1960)
- Le format UNICODE (depuis les années 1990)

Voir les documentations sur le codage de caractères associées à ce cours.

3. Choix d'un type de donnée pour une variable

Les questions à se poser

- ◆ *Pour mettre en place un comptage pouvant aller jusqu'à un nombre maximum de x pièces, quelle taille de variable est nécessaire ?*
- ◆ *Pour gérer une hauteur de 3 mètres avec une précision de $1/10^e$ de mm, quel type de données peut-on utiliser ?*

On sait :

- dans 1 bit, on peut stocker 2 valeurs,
- dans n bits, on peut stocker 2^n valeurs différentes, donc distinguer une valeur pouvant aller de 0 à $2^n - 1$ en nombres non signés (toujours positifs).

Il suffit de choisir le nombre n de bits minimum permettant d'obtenir $2^n > x$

Application

On désire compter un stock de 2000 pièces.

Sur 8 bits : $2^8 = 256$; $256 < 2000$:
insuffisant

Sur 16 bits : $2^{16} = 65536$; $65536 > 2000$:
suffisant.

On choisira donc un mot de 16 bits.

Choisissons une unité de référence : les
« mm ».

Calculons combien de « pas » il faudra distinguer tout au long de notre échelle :

- Hauteur maximum : 3 000 mm
- Hauteur minimum : 0 mm
- Étendue de l'échelle =
maxi - mini = 3000 mm
- Précision : 0,1 mm
- Nombre de pas =
Étendue / précision = $3000 / 0,1 = 30\ 000$
points
- Nombre de bits minimum pour distinguer
30 000 points :
15 bits car $2^{15} = 32768$
- Type de données à utiliser :
Type entier sur 16 bits « Word », signé ou non

- ◆ *Pour gérer une température variant de -200°C à +850°C avec une précision de 0,02°C, quel type de données peut-on utiliser ?*

4. Les calculs numériques sur API

Limitations liées au matériel

Étant données les limitations de stockage de valeurs liées aux formats des données, il faudra s'assurer, **à chaque calcul, final ou intermédiaire**, que le résultat entre bien dans les limites disponibles.

Ainsi, on peut être surpris parfois par des résultats obtenus.

Exemple

	s
28000	0110110101100000
+ 7000	<u>0001101101011000</u>
=	1000100010111000
	↑ Signe passé à 1, donc résultat négatif
Interprétation du résultat en décimal, en calculant l'opposé :	
	1000100010111000
Inversion	0111011101000111
+1	_____1
=	0111011101001000 = 30536 ₍₁₀₎
Le résultat de l'addition 28000 + 7000 serait donc - 30536 !!	

Description de systèmes automatisés séquentiels par GRAFCET

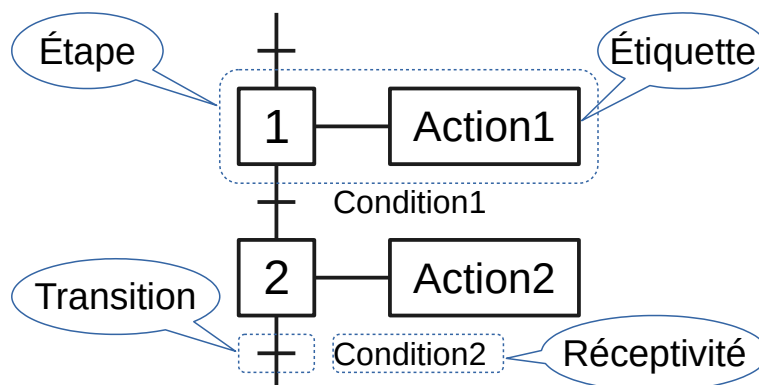
1. Description générale

La création d'une machine automatisée nécessite un **dialogue entre le client** qui définit le cahier des charges (qui contient les besoins et les conditions de fonctionnement de la machine) **et le constructeur qui propose des solutions**. Ce dialogue n'est pas toujours facile : le client ne possède peut-être pas la technique lui permettant de définir correctement son problème. D'autre part, le langage courant ne permet pas de lever toutes les ambiguïtés dues au fonctionnement de la machine (surtout si des actions doivent se dérouler simultanément). C'est pourquoi l'**ADEPA** (Agence pour le Développement de la Productique Appliquée à l'industrie) a créé le **GRAFCET**.

1.1. Définition

Le GRAFCET (GRAphe Fonctionnel de Commande des étapes et Transitions) est l'outil de représentation graphique d'un cahier des charges. Il a été proposé par l'ADEPA en 1977 et normalisé en 1982 par la NF C03-190, remplacé en 2013 par la norme NF EN 60848

Le GRAFCET est une représentation alternée d'**étapes** et de **transitions**. Une seule transition doit séparer deux étapes.



Une **étape** correspond à une situation dans laquelle les variables de sorties conservent leur état. Les **actions** associées aux étapes sont inscrites dans les **étiquettes**.

Une **transition** indique la possibilité d'évolution entre deux étapes successives. A chaque transition est associée une condition logique appelée **réceptivité**.


1.2. Ressources

- Merci à Patrick Abati pour son [Cours sur le Grafcet](#)
- Bibliographie : [E.PEULOT et S. MORENO](#)
- [Norme AFNOR NF EN 60848](#)

1.3. Règles de syntaxe

❖ L'étape initiale : 

- est automatiquement **activée à la mise sous tension** de la partie commande,
- est identifiée avec une case de numéro à **double bordure**,
- est en général une étape d'attente, sans étiquette d'action associée,
- représente la **situation à l'instant initial**, choisie par le concepteur.

❖ Une étape d'action : 

- est *identifiée* par un **numéro d'étape** encadré à gauche, **unique** dans l'ensemble du Grafcet, mais sans signification particulière, les numéros n'ont pas obligation de se suivre ;
- a un numéro d'étape relié aux actions par un trait horizontal à mi-hauteur ;
- peut comporter **une ou plusieurs actions** simultanées, une action par cadre, les cadres sont contigus (ils se touchent).

❖ Les actions :

- s'identifient par des **verbes** d'actions à l'infinitif, *au choix selon la partie opérative* ;
- sont représentées par des symboles ou expressions courtes et facilement identifiables :

TG :	Déplacer en translation gauche	RH :	Rotation dans le sens horaire
TD :	Déplacer en translation à droite	RA :	Rotation dans le sens anti-horaire
TH :	Déplacer vers le haut	A+ :	Sortir le vérin libellé « A »
TB :	Déplacer vers le bas	A- :	Rentrer le vérin libellé « A »
		B :	Activer la broche

❖ Les réceptivités :

- doivent renvoyer une réponse de type binaire : **Oui / Non**, Vrai / Faux, Actif / Non actif,
- sont définies par des **conditions** simples ou des **opérations logiques** de conditions : ET, OU :
 - S5 : si le bouton-poussoir S5 est activé, lorsqu'on appuie sur S5,
 - $\overline{S3}$: si le bouton-poussoir S3 n'est pas activé,
 - DP . S2 : si le détecteur de présence pièce DP est activé ET que le bouton-poussoir S2 est activé **en même temps**,
 - BP + FC : si le bouton-poussoir est activé OU que le fin de course est atteint.

1.4. Règles d'évolution

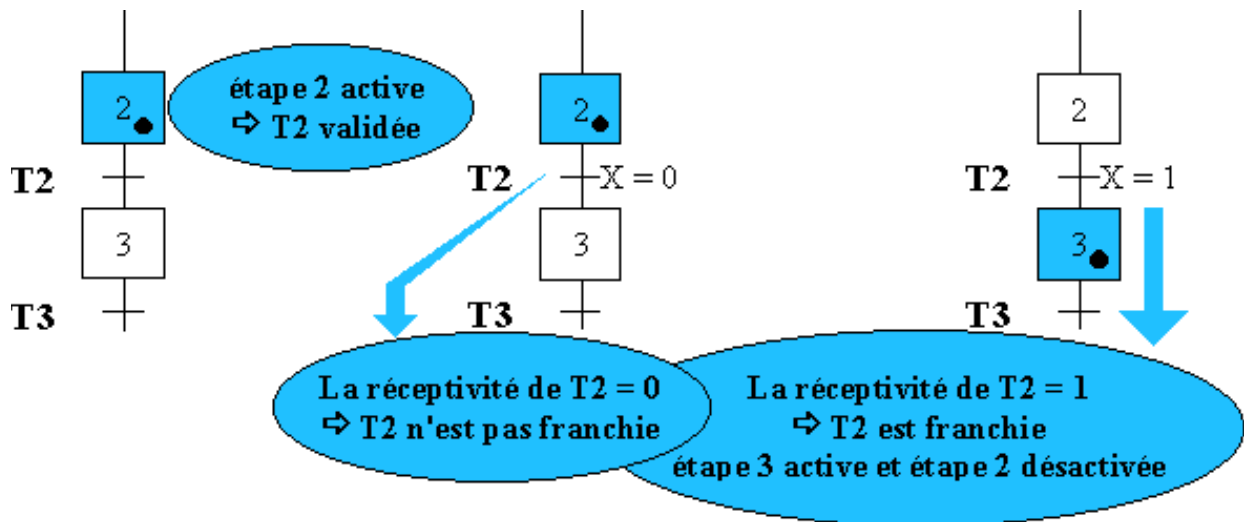
❖ Franchissement d'une transition

Une transition est **franchie** lorsque l'étape associée est **active** et la **réceptivité** associée à cette transition est **vraie**.

❖ Evolution des étapes actives

Le franchissement d'une transition provoque simultanément :

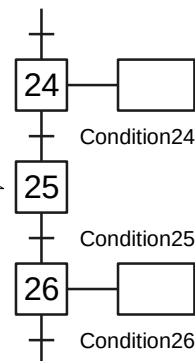
- la **désactivation** de toutes les étapes immédiatement précédentes reliées à cette transition,
- l'**activation** de toutes les étapes immédiatement suivantes reliées à cette transition.



❖ Le système est cyclique : la séquence est bouclée, une étape ne finit jamais « dans le vide ».

❖ Les « étapes d'attente » permettent d'attendre une réceptivité (conditions) sans action liée.

Étapes d'attente sans action

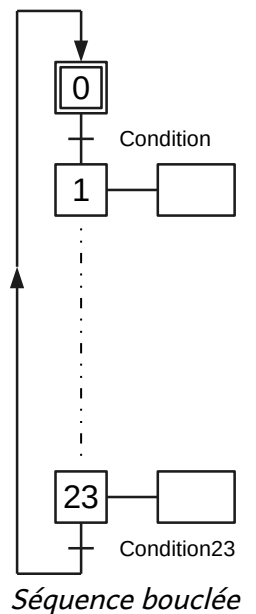


❖ Transitions simultanées

- Plusieurs transitions simultanément franchissables sont simultanément franchies.

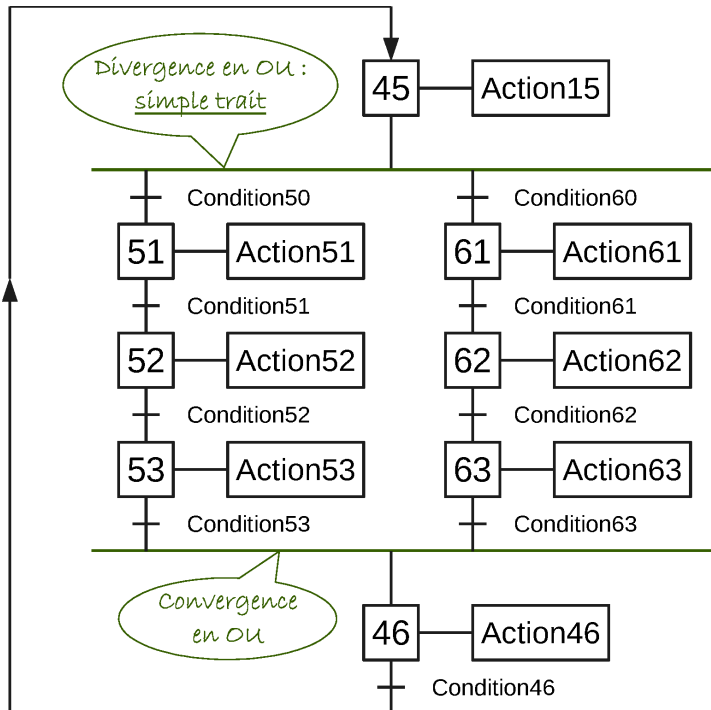
❖ Activation et désactivation simultanées

- Une étape à la fois activée et désactivée reste active.



2. Structures de base

2.1. Divergence et convergence en OU (aiguillage)



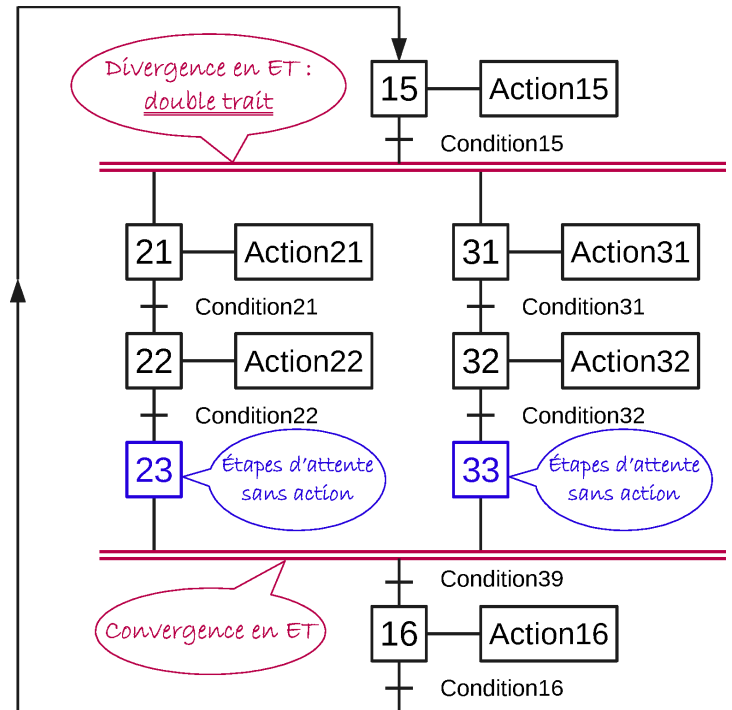
Divergence en OU : l'évolution du système vers une branche dépend des réceptivités Condition50 et Condition60 associées aux transitions.

Convergence en OU : après l'évolution dans une branche, il y a convergence vers une étape commune.

● REMARQUES :

- A et B ne peuvent être vrais simultanément (conflit).
- Après une divergence en OU, on trouve une convergence en OU.
- Le nombre de branches peut-être supérieur à 2.
- La convergence de toutes les branches ne se fait pas obligatoirement au même endroit.

2.2. Divergence et convergence en ET (séquences simultanées)



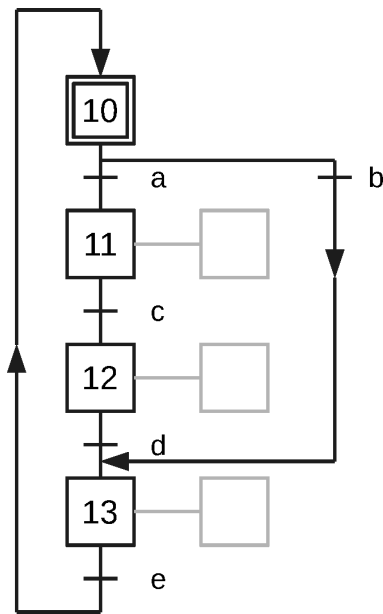
Divergence en ET : lorsque la transition15 est franchie, les étapes 21 et 31 sont actives.

Convergence en ET : la transition39 sera validée lorsque les étapes 23 et 33 seront actives. Si la réceptivité associée à cette transition est vraie, alors celle-ci est franchie.

❖ REMARQUES :

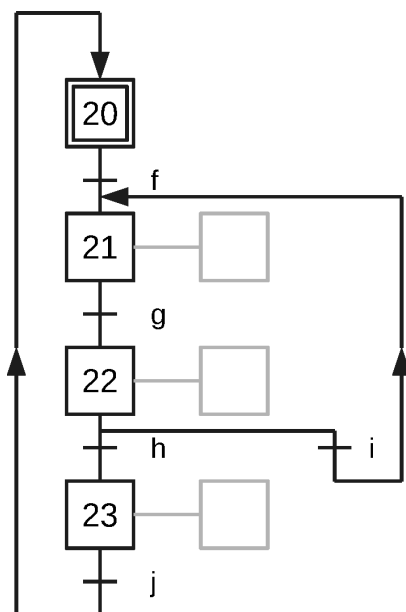
- Après une divergence en ET, on trouve une **convergence en ET**.
- Chaque branche se termine par une **étape d'attente**.
- Le nombre de branches parallèles peut-être supérieur à 2.
- La **réceptivité associée à la convergence** peut-être de la forme « = 1 ». Dans ce cas la transition est franchie dès qu'elle est active.

2.3. Saut en avant (saut de phase)



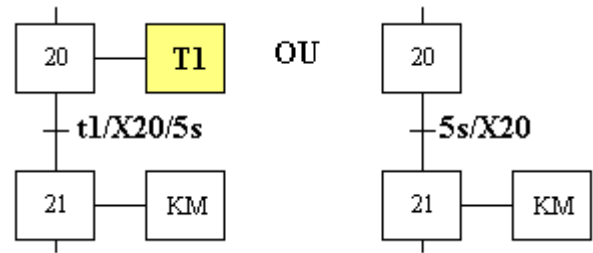
Le saut en avant permet de sauter une ou plusieurs étapes lorsque les actions à réaliser deviennent inutiles.

2.4. Saut en arrière (reprise de phase)



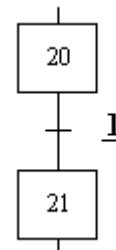
Le saut en arrière permet de reprendre une séquence lorsque les actions à réaliser sont répétitives.

2.5. Temporisations

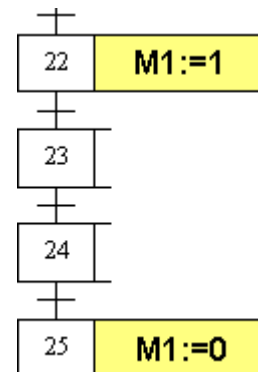


La transition 20 - 21 est franchie lorsque la temporisation, démarrée à l'étape 20 est écoulée, soit au bout de 5s.

2.6. Réceptivité toujours vraie

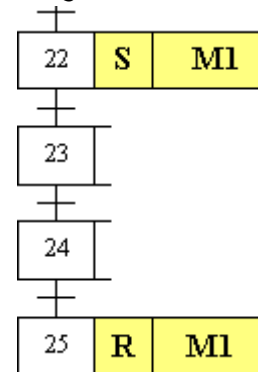


2.7. Action mémorisée



L'action M1 est active aux étapes 22, 23 et 24.

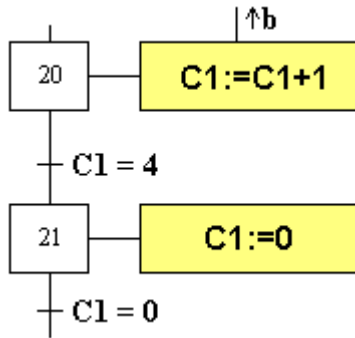
- Sur certains logiciels :



Mise à 1 par S (set), mise à 0 par R (reset)

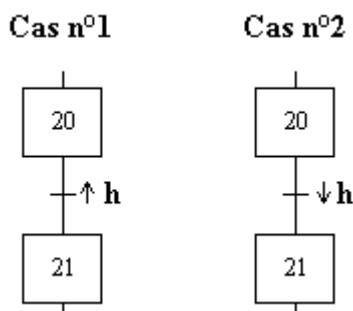
3. Fonctions avancées

3.1. Comptage



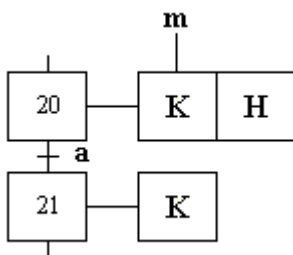
- La transition 20 - 21 est franchie lorsque le contenu du compteur C1 est égal à 4.
- Le compteur est incrémenté sur front montant du signal b.
- Il est mis à zéro à l'étape 21.

3.2. Évènements (fronts)



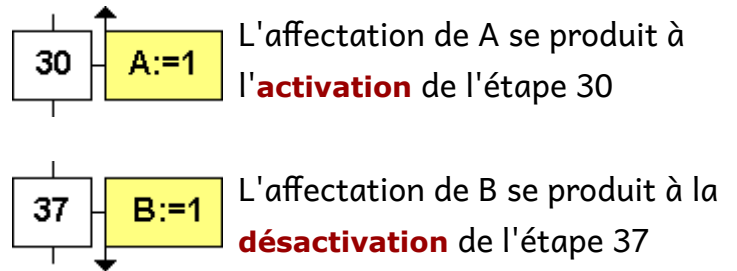
La transition 20 - 21 est franchie lors d'un **front montant** sur h (cas n°1), ou lors d'un **front descendant** sur h (cas n°2).

3.3. Action conditionnelle



L'action K devient effective à l'étape 20, lorsque la condition m est vraie.

3.4. Action à l'activation ou à la désactivation



3.5. Macros

