Automates programmables Twido

Guide de mise en œuvre logicielle

TWD USE 10AF Version 1.0

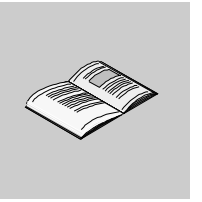
Tome 2

Mai 2002

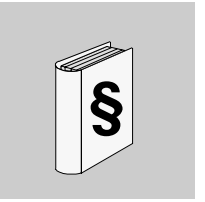
Telemecanique

Schneider Electric

Table des matières



Consignes de sécurité



Informations importantes

AVIS

Lisez attentivement ces instructions et familiarisez-vous avec le matériel avant d'essayer d'installer le périphérique, de le faire fonctionner ou d'effectuer une opération de maintenance. Les messages spéciaux qui suivent peuvent apparaître partout dans ce document ou sur l'appareil. Ils vous avertissent de dangers potentiels ou attirent votre attention sur des renseignements pouvant éclairer ou simplifier une procédure.

La présence de ce symbole sur une étiquette de danger ou d'avertissement indique qu'un risque d'électrocution existe, pouvant provoquer des lésions corporelles si les instructions ne sont pas respectées.

Ceci est le symbole d'une alerte de sécurité. Il sert à vous avertir d'un danger potentiel de blessures corporelles. Respectez toutes les consignes de sécurité accompagnant ce symbole pour éviter toute situation potentielle de blessure ou de mort.

DANGER

La mention DANGER signifie qu'il existe une situation potentiellement dangereuse qui, si elle n'est pas évitée, **entraînera** la mort, des blessures graves ou des dommages matériels.

AVERTISSEMENT

La mention AVERTISSEMENT signifie qu'il existe une situation potentiellement dangereuse qui, si elle n'est pas évitée, **peut entraîner** la mort, des blessures graves ou des dommages matériels.

ATTENTION

La mention ATTENTION signifie qu'il existe une situation potentiellement dangereuse qui, si elle n'est pas évitée, **peut entraîner** des lésions corporelles ou des dommages matériels.

VEUILLEZ REMARQUER

L'entretien du matériel électrique ne doit être effectué que par du personnel qualifié. Schneider Electric n'assume aucune responsabilité des conséquences éventuelles découlant de l'utilisation de cette documentation. Ce document n'est pas destiné à servir de manuel d'utilisation aux personnes sans formation. Le manuel de référence du matériel Twido, TWD USE 10AF, contient les instructions d'assemblage et d'installation.

Informations supplémentaires relatives à la sécurité

Les personnes chargées de l'application, de la mise en oeuvre ou de l'utilisation de ce produit doivent s'assurer que les principes de conception nécessaires ont été inclus dans chacune des applications, en totale conformité avec les normes, codes, régulations, exigences en matière de performance et de sécurité et lois en vigueur.

Avertissements généraux et précautions à prendre

AVERTISSEMENT

RISQUE D'EXPLOSION

- Le remplacement de composants risque d'affecter la conformité de l'équipement à la Classe 1, Division 2.
- Assurez-vous que l'alimentation est coupée ou que la zone ne présente aucun danger avant de déconnecter l'équipement.

Le non-respect de ces précautions peut entraîner des lésions corporelles graves ou/et des dommages matériels importants.

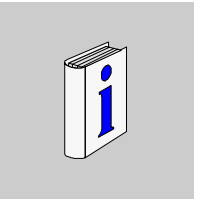
AVERTISSEMENT

FONCTIONNEMENT ACCIDENTEL DE L'EQUIPEMENT

- Coupez l'alimentation avant de procéder à tout retrait, installation, câblage, entretien et contrôle.
- Ce produit n'est pas conçu pour être utilisé lors d'opérations dangereuses pour la sécurité. Lorsque des risques de lésions corporelles ou de dommages matériels existent, utilisez les verrous de sécurité câblés appropriés.
- Ne pas désassembler, réparer ou modifier les modules.
- Cet automate est conçu pour être utilisé dans un boîtier.
- Installez les modules dans des conditions de fonctionnement normales.
- L'alimentation des capteurs ne doit servir qu'à alimenter les capteurs connectés au module.
- Utilisez un fusible approuvé IEC60127 sur la ligne électrique et le circuit de sortie pour satisfaire aux exigences de tension et de courant. Fusible recommandé : Littelfuse 5x20 mm de type slowblow série 218000/Type T.

Le non-respect de ces précautions peut entraîner des lésions corporelles graves ou/et des dommages matériels importants.

A propos de ce manuel



Présentation

Objectif du document

Le manuel de référence du logiciel des automates programmables Twido est composé des sections suivantes :

- Description du logiciel de programmation Twido et introduction aux notions fondamentales requises pour programmer les automates Twido.
- Description des communications, de la gestion des E/S analogiques et d'autres fonctions spéciales.
- Description des langages logiciels utilisés pour créer des programmes Twido.
- Description des instructions et des fonctions des automates Twido.

Champ d'application

Les informations de ce manuel sont **uniquement** destinées aux automates Twido programmables.

Avertissements liés au(x) produit(s)

Schneider Electric décline toute responsabilité pour toute erreur susceptible de figurer dans ce document. Toute reproduction de ce document, complète ou partielle, sur quelque support que ce soit (y compris sous forme électronique) est formellement interdite sans l'autorisation écrite préalable de Schneider Electric.

Description du logiciel Twido

I

En bref...

Présentation

Cette rubrique fournit une introduction aux langages du logiciel, ainsi que les principales informations requises pour créer des programmes de régulation des automates programmables Twido.

Contenu de cet intercalaire

Cet intercalaire contient les chapitres suivants :

Introduction au logiciel Twido

1

En bref...

Présentation

Ce chapitre offre une introduction rapide à TwidoSoft, le logiciel de programmation et de configuration des automates Twido, ainsi qu'aux langages de programmation Grafcet, liste d'instructions ou schéma à contacts.

Contenu de ce chapitre

Ce chapitre contient les sujets suivants :

Introduction à TwidoSoft

Introduction

TwidoSoft est un environnement de développement graphique permettant de créer, de configurer et de gérer des applications de régulation des automates programmables Twido. TwidoSoft vous permet d'entrer des programmes de régulation à l'aide des éditeurs TwidoSoft de programmes par schémas à contacts ou par listes, puis de transférer le programme en vue de son exécution sur un automate.

TwidoSoft

TwidoSoft est un programme 32 bits pour PC fonctionnant sous Windows 98 Deuxième édition ou sous Windows 2000 Professionnel.
Principales fonctionnalités logicielles offertes par TwidoSoft :

- interface utilisateur Windows standard
- programmation et configuration d'automates Twido
- communication et régulation d'automates

Pour plus d'informations, reportez-vous au guide d'exploitation TwidoSoft.

Introduction aux langages Twido

Introduction

Un automate programmable lit des entrées, génère des sorties et résout une logique basée sur un programme de régulation. La création du programme de régulation d'un automate Twido consiste en l'écriture d'une série d'instructions rédigées dans un des langages de programmation Twido.

Languages Twido

Les langages suivants peuvent être utilisés pour créer des programmes de régulation d'automates Twido :

- Langage liste d'instructions
  Un programme liste d'instructions est constitué d'une série d'expressions logiques, rédigées sous la forme d'une séquence d'instructions booléennes.
- Langage schéma à contacts
  Un schéma à contacts est une représentation graphique d'une expression logique.
- Langage Grafcet
  Twido comprend les instructions liste Grafcet, mais pas les objets de représentation graphique Grafcet.

Les opérations de création et d'édition de programmes de régulation Twido à l'aide de ces langages de programmation peuvent être réalisées depuis un ordinateur personnel (PC).
Une fonctionnalité de réversibilité liste d'instructions / schéma à contacts vous permet de convertir un programme en langage liste d'instructions dans le langage schéma à contacts, et vice-versa.

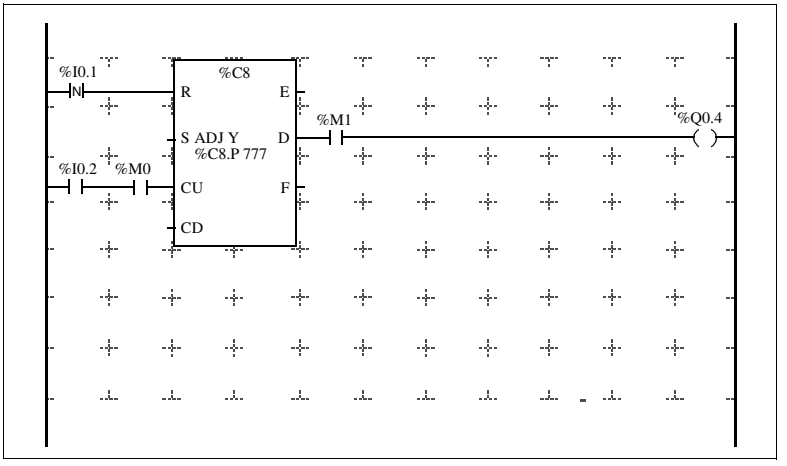
Langage liste d'instructions

Un programme rédigé en langage liste d'instructions consiste en une série d'instructions exécutées de manière séquentielle par l'automate. Vous trouverez ci-dessous un exemple de programme en langage liste d'instructions.

```
0   BLK     %C8
1   LDF     %I0.1
2   R
3   LD      %I0.2
4   AND     %M0
5   CU
6   OUT_BLK
7   LD      D
8   AND     %M1
9   ST      %Q0.4
10  END_BLK
```

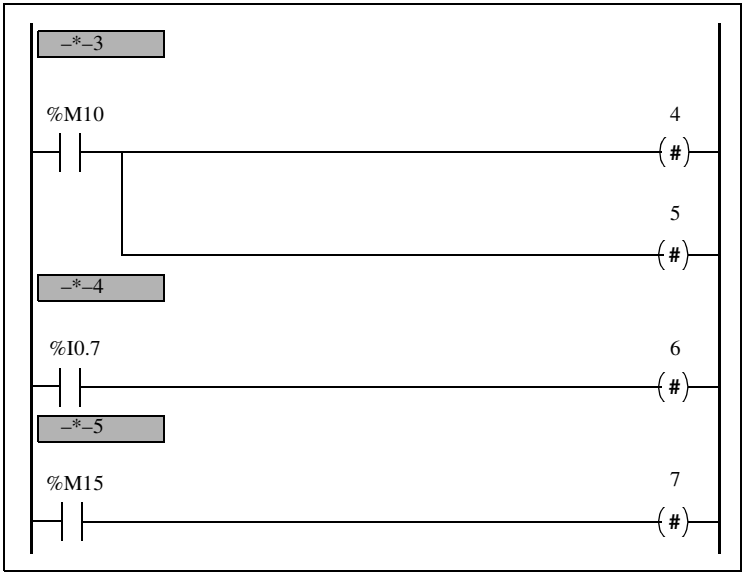
Langage schéma à contacts

Les schémas à contacts utilisent la même représentation graphique que celle des circuits de relais en logique programmée. Dans ces schémas, les éléments graphiques, tels que des bobines, des contacts et des blocs représentent les instructions du programme. Ci-dessous un exemple de schéma à contacts.



Langage Grafcet La méthode analytique Grafcet divise toute régulation d'automatisation en une série d'étapes auxquelles des actions, des transitions et des conditions sont associées. Vous trouverez ci-dessous des exemples d'instructions Grafcet, rencontrées respectivement dans des programmes liste d'instructions et schéma à contacts.

```
0    -*-    3
1    LD     %M10
2    #      4
3    #      5
4    -*-    4
5    LD     %I0.7
6    #      6
7    -*-    5
8    LD     %M15
9    #      7
10   ...
```



Objets langage Twido

2

En bref...

Présentation

Ce chapitre offre une description détaillée des objets langage de programmation des automates Twido.

Contenu de ce chapitre

Ce chapitre contient les sujets suivants :

Validation d’un objet langage

Introduction

Les objets mots et bits ne sont valides que lorsqu'ils ont été alloués à une zone mémoire de l'automate. Pour que cette allocation soit possible, il est nécessaire que ces objets aient été utilisés dans l'application avant d'être téléchargés vers l'automate.

Exemple

La plage d'objets valides est comprise entre 0 et la référence maximum autorisée pour ce type d'objet. Par exemple, si la référence maximum autorisée pour les mots mémoire dans votre application est %MW9, les zones %MW0 à %MW9 sont allouées. Dans cet exemple, %MW10 n'est pas valide. Aucun accès à cette zone n'est autorisé, aussi bien de manière interne qu'externe.

Objets bits

Introduction

Les objets bits sont des bits variable logiciels qui sont de simples bits de données qui peuvent être utilisés comme des opérandes et testés par des instructions booléennes. Vous trouverez ci-dessous la liste des objets bits :

- Bits d'E/S
- Bits internes (bits mémoire)
- Bits système
- Bits étape
- Bits extraits de mots

Liste des bits opérandes

Le tableau suivant répertorie et décrit l'ensemble des objets bits principaux qui sont utilisés comme opérandes dans des instructions booléennes.

Type	Description	Repère ou valeur	Nombre maximal	Accès en écriture[1]
Valeurs immédiates	0 ou 1 (False ou True)	0 ou 1	-	-
Entrées Sorties	Ces bits sont les "images logiques" des états électriques des E/S. Ils sont stockés dans la mémoire de données et sont mis à jour à chaque scrutation de la logique du programme.	%Ix.y.z[2] %Qx.y.z[2]	Note[4]	No Oui
Interne (mémoire)	Les bits internes sont des zones de mémoire internes utilisées pour stocker des valeurs intermédiaires lorsqu'un programme est en cours d'exécution. **Note :** Les bits d'E/S non utilisés ne peuvent pas être employés comme des bits internes.	%Mi	128 TWDLCAA10 DRF, TWDLCAA16 DRF 256 Tous les autres automates	Oui
Système	Les bits système %S0 à %S127 surveillent le bon fonctionnement de l'automate ainsi que la bonne exécution du programme de l'application.	%Si	128	Selon i
Blocs fonctions	Les bits des blocs fonctions correspondent aux sorties des blocs fonctions. Ces sorties peuvent être directement câblées ou exploitées en tant qu'objet.	%TMi.Q, %Ci.P, etc.	Note[4]	Non[3]
Blocs fonctions réversibles	Blocs fonctions programmés à l'aide d'instructions de programmation réversible BLK, OUT_BLK et END_BLK.	E, D, F, Q, TH0, TH1	Note[4]	Non
Extraits de mots	Pour certains mots, un des 16 bits est extrait en tant que bit opérande.	Variable	Variable	Variable

Type	Description	Repère ou valeur	Nombre maximal	Accès en écriture[1]
Etapes Grafcet	Les bits %X1 à %Xi sont associés aux étapes Grafcet. Le bit étape Xi est réglé sur 1 lorsque l'étape correspondante est active et sur 0 lorsqu'elle est désactivée.	%X21	62 TWDLCAA10 DRF, TWDLCAA16 DRF 94 TWDLCAA24 DRF, Automates modulaires	Oui

Notes :

1. Ecrit par le programme ou à l'aide de l'éditeur de table d'animation.

2. Reportez-vous à la section "Repérage des E/S".

3. Ces bits, à l'exception de %SBRi.j et de %SCi.j, sont accessibles en écriture et en lecture.

4. Ce nombre est déterminé par le modèle de l'automate.

Objets mots

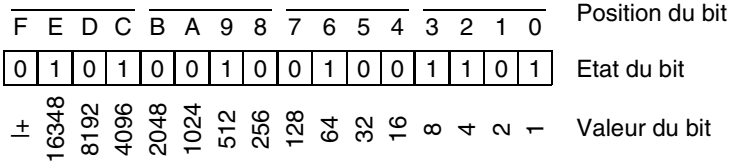
Introduction

Les objets mots sont repérés sous la forme de mots de 16 bits rangés dans la mémoire de données et peuvent contenir un entier compris entre –32 768 et 32 767 (sauf pour le bloc fonction compteur rapide (FC) qui est compris entre 0 et 65 535).
Exemples d'objets mots :

- Valeurs immédiates
- Mots internes (%MWi) (mots mémoire)
- Mots constants (%KWi)
- Mots échanges E/S (%IWi, %QWi)
- Mots système (%SWi)
- Blocs fonctions (données de configuration et/ou d'exécution)

Formats de mot

Le contenu des mots ou des valeurs est rangé dans la mémoire utilisateur sous la forme d'un code binaire à 16 bits (complément à deux) utilisant la convention suivante :



Pour les notations binaires signées, le bit 15 est attribué, par convention, au signe de la valeur codée :

- Le bit 15 est égal à 0 : le mot contient une valeur positive.
- Le bit 15 est égal à 1 : le mot contient une valeur négative (les valeurs négatives sont exprimées en complément à deux).

Il est possible d'entrer et de récupérer les mots et les valeurs immédiates sous les formats suivants :

- Décimal
  Min : -32 768, Max : 32 767 (1579, par exemple)
- Hexadécimal
  Min : 16#0000, Max : 16#FFFF (16#A536, par exemple)
  Syntaxe alternative : #A536

Description des objets mots

Le tableau suivant décrit les objets mots.

Mots	Description	Repère ou valeur	Nombre maximal	Accès en écriture[1]
Valeurs immédiates	Il s'agit d'entiers dont le format est identique à celui des mots de 16 bits. Cela permet d'attribuer des valeurs à ces mots.		-	Non
	Base 10	-32 768 à 32 767		
	Base 16	16#0000 à 16#FFFF		
Interne (mémoire)	Mots utilisés pour ranger des valeurs dans la mémoire des données au cours du fonctionnement. Les mots %MWO à %MW255 sont directement lus et écrits par le programme.	%MWi	1500	Oui
Constants	Mémorisent les constantes ou les messages alphanumériques. Leur contenu ne peut être écrit ou modifié qu'en utilisant TwidoSoft au cours de la configuration. Le programme ne peut accéder aux mots constants de %KW0 à %KW63 qu'en lecture.	%KWi	64	Oui, uniquement à l'aide de TwidoSoft
Système	Ces mots de 16 bits proposent plusieurs fonctions : • Ils permettent l'accès aux données provenant directement de l'automate en lisant les mots %SWi (potentiomètres, par exemple). • Ils effectuent des opérations sur l'application (l'ajustement des blocs horodateurs, par exemple).	%SWi	128	Selon i
Blocs fonctions	Ces mots correspondent aux paramètres ou aux valeurs courantes des blocs fonctions.	%TM2.P, %Ci.P, etc.		Oui

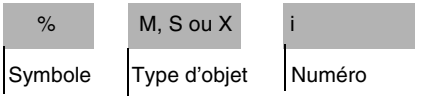
Mots	Description	Repère ou valeur	Nombre maximal	Accès en écriture[1]
Mots échanges E/S	Attribués aux automates connectés en tant que Liaisons distantes. Ces mots sont utilisés pour la communication entre les automates.			
	Entrées	%IWi.j	Note[2]	Non
	Sorties	%QWi.j	Note[2]	Oui
Bits extraits	Il est possible d'extraire un des 16 bits à partir des mots suivants :			
	Interne	%MWi:Xk	1500	Oui
	Système	%SWi:Xk	128	Dépend de i
	Constants	%KWi:Xk	64	Non
	Entrée	%IWi.j:Xk	Note[2]	Non
	Sortie	%QWi.j:Xk	Note[2]	Oui

Note :
1. Ecrit par le programme ou à l'aide de l'éditeur de table d'animation.
2. Ce nombre est déterminé par le modèle de l'automate.

Adressage d’objets bits

Syntaxe

L’adressage des objets bits d’étape, internes et système doit se conformer à la syntaxe suivante :



Description

Le tableau suivant décrit les éléments de la syntaxe d'adressage.

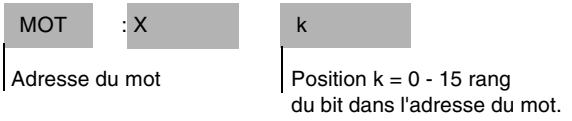
Groupe	Elément	Description
Symbole	%	Une variable logicielle doit toujours débuter par un symbole de pourcentage (%).
Type d’objet	M	Les bits internes permettent de stocker des valeurs intermédiaires lorsqu'un programme est en cours d'exécution.
	S	Les bits système offrent des informations d'état et de régulation relatives à l'automate.
	X	Les bits d’étape offrent des informations sur l'état des activités des étapes.
Numéro	i	La valeur maximum dépend du nombre d’objets configurés.

Exemples d’adressage d’objets bits :

- %M25 = bit interne numéro 25
- %S20 = bit système numéro 20
- %X6 = bit étape numéro 6

Objets bits extraits de mots

TwidoSoft permet d'extraire un des 16 bits des mots. L'adresse du mot est alors complétée par le rang du bit extrait suivant la syntaxe suivante.



Exemples :

- %MW5:X6 = bit numéro 6 du mot interne %MW5
- %QW5.1:X10 = bit numéro 10 du mot de sortie %QW5.1

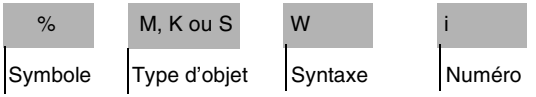
Adressage d’objets mots

Introduction

L'adressage d'objets mots doit se conformer à la syntaxe décrite ci-dessous. Veuillez noter que cette syntaxe ne s'applique PAS à l'adressage d'E/S (reportez-vous à la rubrique *Repérage des entrées/sorties, p. 33*) et des blocs fonctions (reportez-vous à la rubrique *Objets blocs fonctions, p. 36*).

Syntaxe

L’adressage des mots internes, constants et système doit se conformer à la syntaxe suivante.



Description

Le tableau suivant décrit les éléments de la syntaxe d'adressage.

Groupe	Elément	Description
Symbole	%	Une adresse interne doit toujours débuter par un symbole de pourcentage (%).
Type d’objet	M	Les mots internes permettent de stocker des valeurs intermédiaires lorsqu'un programme est en cours d'exécution.
	K	Les mots constants permettent de stocker des valeurs constantes ou des messages alphanumériques. Leur contenu ne peut être écrit ou modifié qu'en utilisant TwidoSoft.
	S	Les mots système offrent des informations d'état et de régulation relatives à l'automate.
Syntaxe	W	Mot de 16 bits.
Numéro	i	La valeur maximum dépend du nombre d’objets configurés.

Exemples d’adressage d’objets mots :

- %MW15 = mot interne numéro 15
- %KW26 = mot constant numéro 26
- %SW30 = mot système numéro 30

Repérage des entrées/sorties

Introduction

Chaque point d'E/S (entrée/sortie) d'une configuration Twido possède un repère unique. Par exemple, le repère "%I0.0.4" peut être affecté à une entrée spécifique d'un automate.
Des repères d'E/S peuvent être affectés aux matériels suivants :

- Automate configuré en tant que maître de liaison distante
- Automate configuré en tant qu'E/S distante
- Modules d'E/S d'expansion

Références multiples à une sortie ou à une bobine

Un programme peut comporter plusieurs références à une même sortie ou bobine. Seul le résultat de la dernière référence traitée est mis à jour au niveau des sorties du matériel. Par exemple, %Q0.0.0 peut être utilisé plusieurs fois dans un programme sans qu'un avertissement ne signale la multiplicité des occurrences. Il est donc important de confirmer quelle sortie provoquera l'opération souhaitée.

ATTENTION

Opération inattendue

Les doublons de sortie ne sont pas contrôlés et aucun avertissement n'est donné. Vérifiez l'utilisation qui est faite des sorties et des bobines avant de les modifier dans l'application.

Le non-respect de ces précautions peut entraîner des lésions corporelles ou/et des dommages matériels.

Syntaxe

Le repérage des entrées et des sorties doit se conformer à la syntaxe suivante.

%	I, Q	x	y	z
Symbole	Type d'objet	Position de l'automate	Type d'E/S	Numéro de voie

Description

Le tableau suivant décrit la syntaxe de repérage des E/S.

Groupe	Elément	Valeur	Description
Symbole	%	-	Un repère interne doit toujours débuter par un symbole de pourcentage (%).
Type d'objet	I	-	Entrée. "Image logique" de l'état électrique de l'entrée d'un automate ou d'un module d'E/S d'expansion.
	Q	-	Sortie. "Image logique" de l'état électrique de la sortie d'un automate ou d'un module d'E/S d'expansion.
Position de l'automate	x	0 1 - 7	Automate maître (maître de liaison distante). Automate distant (esclave de liaison distante).
Type d'E/S	y	0 1 - 7	E/S de base (E/S locale sur un automate). Modules d'E/S d'expansion.
Numéro de voie	z		Numéro de la voie d'E/S sur l'automate ou le module d'E/S d'expansion. Le nombre de points d'E/S disponibles dépend du modèle de l'automate ou du type du module d'E/S d'expansion.

Exemples

Le tableau suivant présente quelques exemples de repérage des E/S.

Objet d'E/S	Description
%I0.0.5	Point d'entrée n° 5 sur la base automate (E/S locale).
%Q0.3.4	Point de sortie n° 4 sur le module d'E/S d'expansion au repère d'expansion n° 3 pour la base automate (E/S d'expansion).
%I0.0.3	Point d'entrée n° 3 sur la base automate.
%I3.0.1	Point d'entrée n° 1 sur l'automate d'E/S distant au repère de liaison distante n° 3.
%I0.3.2	Point d'entrée n° 2 sur le module d'E/S d'expansion au repère n° 3 pour la base automate.

Adressage réseau

Introduction

Les mots réseau %INW et %QNW permettent d'échanger des données d'application entre les automates d'extension et l'automate maître sur un réseau de liaison distante Twido. Reportez-vous à la rubrique *Communications , p. 69* pour obtenir plus d'informations.

Syntaxe

L'adressage réseau doit se conformer à la syntaxe suivante.

Description de la syntaxe

Le tableau suivant décrit la syntaxe d'adressage réseau.

Groupe	Elément	Valeur	Description
Symbole	%	-	Une adresse interne doit toujours débuter par un symbole de pourcentage (%).
Type d'objet	IN	-	Mot d'entrée réseau. Transfert de données de l'automate maître vers l'automate d'extension.
	QN	-	Mot de sortie réseau. Transfert de données de l'automate d'extension vers l'automate maître.
Syntaxe	W	-	Mot de 16 bit.
Position de l'automate	x	0 1 - 7	Automate maître (maître de liaison distante). Automate distant (esclave de liaison distante).
Mot	j	0 - 3	Chaque automate d'extension utilise un maximum de quatre mots pour assurer l'échange de données avec l'automate maître.

Exemples

Le tableau suivant présente quelques exemples d'adressage réseau.

Objet réseau	Description
%INW3.1	Mot réseau n°1 de l'automate distant n°3.
%QNW0.3	Mot réseau n°3 de la base automate.

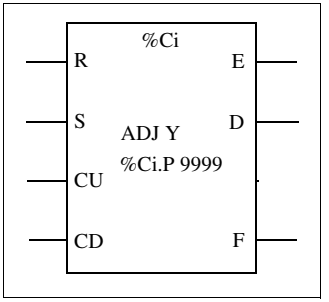
Objets blocs fonctions

Introduction

Les blocs fonctions contiennent des objets bits et des mots spécifiques accessibles par le programme.

Exemple de bloc fonction

L'illustration suivante présente un bloc fonction compteur.



Bloc compteur/décompteur

Objets bits

Les objets bits correspondent aux sorties des blocs. Les instructions booléennes de test peuvent accéder à ces bits selon l'une ou l'autre de ces méthodes :

- directement (LD E, par exemple) s'ils sont liés au bloc par une programmation réversible (reportez-vous à la rubrique *Principes de programmation de blocs fonctions élémentaires, p. 224*).
- en spécifiant le type de bloc (LD %Ci.E, par exemple).

Les instructions peuvent accéder aux entrées.

Objets mots

Les objets mots correspondent aux paramètres et valeurs spécifiés suivants :

- Paramètres de configuration des blocs : le programme peut accéder à certains paramètres (paramètres de présélection, par exemple), mais pas à d'autres (base temps, par exemple).
- Valeurs courantes : %Ci.V, la valeur de comptage courante, par exemple.

Objets accessibles par le programme

Reportez-vous aux rubriques suivantes pour connaître les listes des objets accessibles par le programme.

- Pour les blocs fonctions élémentaires, reportez-vous à la rubrique *Blocs fonctions élémentaires, p. 222*.
- Pour les blocs fonctions avancés, reportez-vous à la rubrique *Objets mots et objets bits associés à des blocs fonctions avancés, p. 273*.

Objets structurés

Introduction

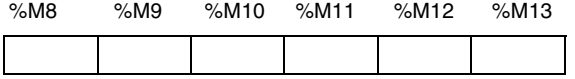
Les objets structurés sont des ensembles formés par des objets simples. Twido prend en charge les types d'objets structurés suivants :

- chaînes de bits
- tables de mots

Chaînes de bits

Les chaînes de bits sont composées d'une série de bits objets adjacents du même type et dont la longueur (L) est définie.

Exemple : Chaîne de bit %M8:6



Note : %M8:6 est acceptable (car 8 est un multiple de 8), alors que %M10:16 ne l'est pas (10 n'est pas un multiple de 8).

Les chaînes de bits peuvent être utilisées avec l'instruction d'affectation (voir section *Instructions d'affectation, p. 248*).

Types de bits disponibles

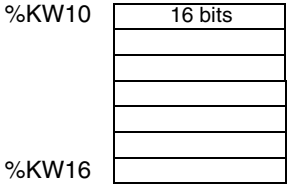
Types de bits disponibles pour les chaînes de bits :

Type	Repère	Taille maximale	Accès en écriture
Bits d'entrée TOR	%I0.0:L ou %I1.0:L1	0<L<17	Non
Bits de sortie TOR	%Q0.0:L ou %Q1.0:L1	0<L<17	Oui
Bits système	%Si:L où "i" est multiple de 8	0<L<17 et i+L-128	En fonction de i
Bits étape Grafcet	%Xi:L où "i" est multiple de 8	0<L<17 et i+L-95	Oui (par programme)
Bits internes	%Mi:L où "i" est multiple de 8	0<L<17 et i+L-256	Oui

Note : (1) Seuls les bits 0...L-1 peuvent être repérés. Toutes les E/S peuvent ne pas être repérées en chaîne bits.

Tables de mots

Les tables de mots sont composées d'une série d'objets adjacents du même type et dont la longueur (L) est définie.
Exemple : Table de mots %KW10:7



Les tables de mots peuvent être utilisées avec l'instruction d'affectation (voir section *Instructions d'affectation, p. 248*).

Types de mots disponibles

Types de mots disponibles pour les tables de mots :

Type	Repère	Taille maximale	Accès en écriture
Mots internes	%MWi:L	0<L<256 et i+L< ou = 1500	Oui
Mots constants	%KWi:L	0<L et i+L-64	Non
Mots système	%SWi:L	0<L et i+L-128	En fonction de i

Mots indexés

Introduction

Un mot indexé est un mot interne ou un mot constant possédant un repère d'objet indexé. Il existe deux types de repérage d'objet :

- repérage direct
- repérage indexé

Repérage direct

Le repère direct d'un objet est défini au moment de l'écriture du programme.
Exemple : %M26 est un bit interne dont le repère direct est 26.

Repérage indexé

L'indexation du repère d'un objet permet de modifier le repère en attribuant un index au repère direct d'un objet. Le contenu de l'index est ajouté au repère direct de l'objet. L'index est défini par un mot interne %MWi. Le nombre de "mots indexés" est illimité.
Exemple : %MW108[%MW2] est un mot dont le repère est composé du repère direct 108 et du contenu du mot %MW2.
Si la valeur du mot %MW2 est 12, le fait d'écrire dans %MW108[%MW2] équivaut à écrire dans %MW120 (108 + 12).

Mots disponibles pour le repérage indexé

Vous trouverez ci-dessous les types de mots disponibles pour le repérage indexé.

Type	Repère	Taille maximale	Accès en écriture
Mots internes	%MWi[MWi]	0-i< ou = %MWj<1500	Oui
Mots constants	%KWi[%MWj]	0-i<%MWj<64	Non

Les mots indexés peuvent être utilisés avec les instructions d'affectation (voir *Instructions d'affectation, p. 248*) et dans les instructions de comparaison (voir *Instructions de comparaison, p. 252*). Ce type de repérage permet de scruter individuellement un ensemble d'objets du même type (tels que des mots internes ou des constantes), en modifiant le contenu du mot indexé via le programme.

Bit système débordement d'index %S20

Un débordement d'index se produit lorsque le repère d'un objet indexé dépasse les limites de la zone mémoire contenant le même type d'objet. Pour résumer :

- Le repère de l'objet plus le contenu de l'index sont inférieurs à 0.
- La somme de l'adresse de l'objet et du contenu de l'index est supérieure au mot le plus grand directement référencé dans l'application. Le nombre maximum est 1499 (pour les mots %MWi) ou 63 (pour les mots %Kwi).

En cas de débordement d'index, le système règle le bit système %S20 sur 1 et une valeur d'index égale à 0 est affectée à l'objet.

Note : L'utilisateur est responsable du contrôle des débordements. Le bit %S20 doit être lu par un programme utilisateur pour rendre le traitement possible. L'utilisateur doit confirmer qu'il est remis à 0.
%S20 (état initial = 0) :

- Sur débordement d'index : réglé sur 1 par le système.
- Constatation de débordement : réglé sur 0 par l'utilisateur, après modification de l'index.

Symbolisation d'objets

Introduction

Les symboles permettent d'adresser des objets du langage logiciel Twido, à l'aide de noms ou de mnémoniques personnalisés. L'utilisation de symboles permet d'examiner et d'analyser rapidement la logique d'un programme et simplifie significativement les procédures de développement et de test d'une application.

Exemple

Par exemple, le symbole WASH_END pourrait être utilisé pour identifier un bloc fonction horodateur correspondant à la fin d'un cycle de lavage. L'utilisation de ce nom se révélera beaucoup plus pratique que celui du repère du programme, tel que %TM3.

Instructions pour la définition de symboles

Les noms de symboles doivent répondre aux exigences suivantes :

- Ces noms doivent comporter un maximum de 32 caractères.
- Ces noms peuvent uniquement comporter des lettres (A-Z), des nombres (0 -9) et des traits de soulignement (_).
- Le premier caractère de ces noms doit être alphanumérique ou accentué. Ces noms ne peuvent pas comporter de signe de pourcentage (%).
- Ces noms ne peuvent pas contenir d'espaces ou de caractères spéciaux.
- Aucune distinction ne sera faite entre les majuscules et les minuscules. Par exemple, "Pompe1" et "POMPE1" correspondront au même symbole et ne pourront par conséquent être utilisés qu'une seule fois dans l'application.

Edition des symboles

Utilisez l'éditeur de symboles pour définir et associer des objets de langage. Il est important de signaler que les symboles et leurs commentaires ne sont pas stockés sur l'automate, mais avec l'application, sur le disque dur. Il est donc impossible de transférer ces symboles vers l'automate, avec l'application.
Reportez-vous au guide d'exploitation TwidoSoft pour plus d'informations sur l'utilisation des symboles.

Mémoire utilisateur

3

Structure de la mémoire utilisateur

Introduction

La mémoire de l'automate accessible par une application utilisateur est divisée en deux ensembles distincts :

- des valeurs de bits
- des valeurs de mots (valeurs signées à 16 bits)

Mémoire bit

La mémoire bit est située dans la mémoire RAM interne intégrée à l'automate. Elle contient l'image des 1280 objets bits.

Rôle de la mémoire mots

La mémoire mots (16 bits) prend en charge :

- **les données** : données dynamiques de l'application et données système
- **le programme** : descripteurs et code exécutable des tâches
- **les constantes** : mots constants, valeurs initiales et configuration des entrées/sorties

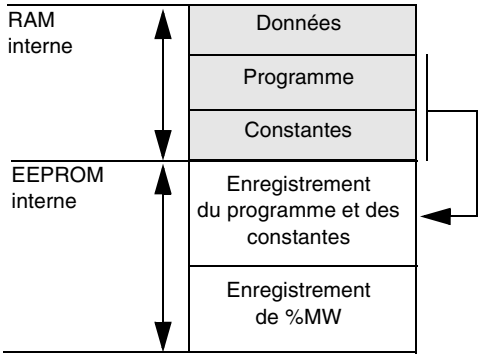
Types de mémoire

Les trois types de mémoire pour les automates Twido sont les suivants :

- RAM interne (intégrée)
  Mémoire RAM intégrée à l'automate. Les 10 premiers Ko de la mémoire RAM interne sont de la RAM rapide et les 32 Ko suivants sont de la RAM standard. La RAM interne contient le programme, les constantes et les données.
- EEPROM interne
  Mémoire EEPROM intégrée de 32 Ko permettant une sauvegarde interne dans l'automate d'une application. Elle protège les applications des altérations causées par un défaut de batterie ou une coupure secteur de plus de 30 jours. Elle contient un programme et des constantes.
- Cartouche de sauvegarde de mémoire externe
  Cartouche EEPROM externe en option pour la sauvegarde d'une application ou l'utilisation d'une application plus importante. Elle peut être utilisée pour mettre à jour l'application dans la mémoire RAM de l'automate. Elle est composée d'un programme et de constantes, mais elle ne contient aucune donnée.

Structure sans cartouche de mémoire externe

Le schéma suivant présente la structure de la mémoire sans cartouche de mémoire externe en option.

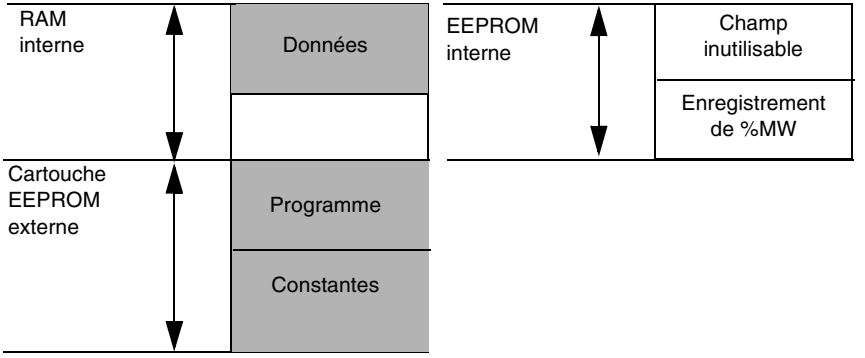


La mémoire EEPROM interne de 32 Ko intégrée à l'automate peut contenir :

- le programme de l'application (32 Ko) et
- 512 mots internes (%MWi)

Structure avec cartouche de mémoire externe

La cartouche de mémoire externe en option permet d'effectuer la sauvegarde des programmes et des constantes. Elle permet également de fournir de la mémoire supplémentaire pour les applications plus importantes.
Le schéma suivant illustre la structure de la mémoire avec cartouche de mémoire externe.



La mémoire EEPROM interne de 32 Ko permet d'enregistrer 512 mots internes (%MWi).

Sauvegarde de la mémoire

La mémoire RAM interne de l'automate est enregistrée par l'un des éléments suivants :

- batterie interne (jusqu'à 30 jours)
- mémoire EEPROM interne (32 Ko maximum)
- cartouche de mémoire externe en option (64 Ko maximum)

Le transfert de l'application depuis la mémoire EPROM interne vers la mémoire RAM s'effectue automatiquement, lorsqu'il y a perte de l'application en RAM (absence de sauvegarde ou de batterie).
Notez qu'il est également possible d'effectuer un transfert manuel à l'aide de TwidoSoft.

Configurations de la mémoire

Le tableau suivant présente les configurations de mémoire possibles des automates Twido.

Type de mémoire	Automates compacts			Automates modulaires		
	10DRF	**16DRF**	**24DRF**	**20DUK 20DTK**	**20DRT**	**40DUK 40DTK**
RAM interne	10 Ko	32 Ko	32 Ko	32 Ko	32 Ko	32 Ko
Mémoire étendue disponible*					64 Ko	64 Ko
Taille maximale de l'application	10 Ko	32 Ko	32 Ko	32 Ko	32 Ko ou 64 Ko*	32 Ko ou 64 Ko*
Sauvegarde externe maximale	32 Ko	32 Ko	32 Ko	64 Ko	32 Ko ou 64 Ko	32 Ko ou 64 Ko

Note : *La mémoire peut être étendue jusqu'à 64 Ko pour les automates TWDLMDA20DRT, TWDLMDA40DUK et TWDLMDA40DTK en installant la cartouche de mémoire externe de 64 Ko en option. La cartouche doit rester installée afin de permettre l'exécution et la sauvegarde de l'application.

Modes de fonctionnement de l'automate

4

En bref...

Présentation

Ce chapitre offre des informations sur les modes de fonctionnement des automates, ainsi que sur l'exécution cyclique et périodique de programmes. Vous y trouverez également des informations détaillées sur les coupures secteur et les opérations de restauration.

Contenu de ce chapitre

Ce chapitre contient les sujets suivants :

Scrutation cyclique

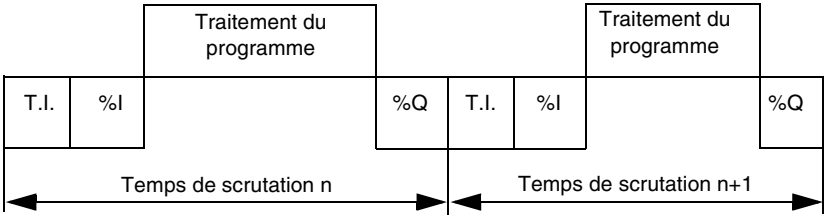
Introduction

La scrutation cyclique relie les cycles de la tâche maître les uns après les autres et n'attend que le traitement du système. Après avoir effectué la mise à jour des sorties (troisième phase du cycle de la tâche), le système exécute un certain nombre de ses propres tâches et déclenche immédiatement un autre cycle de la tâche.

Note : La durée de scrutation du programme utilisateur est contrôlée par le temporisateur chien de garde de l'automate et ne doit pas dépasser 150 ms. Sinon une faute apparaît faisant passer immédiatement l'automate en mode d'arrêt. Sous ce mode, les sorties sont forcées sur leur état de repli par défaut.

Fonctionnement

L'illustration suivante montre la durée des phases d'exécution de la scrutation cyclique.



Description des phases de fonctionnement

Le tableau suivant décrit les phases de fonctionnement.

Repère	Phase	Description
T.I.	Traitement interne	Le système réalise implicitement la surveillance de l'automate (gestion des bits et mots système, mise à jour des valeurs courantes de l'horodateur, mise à jour des voyants d'état, détection des commutateurs RUN/STOP, etc.) et le traitement des requêtes en provenance de TwidoSoft (modifications et animation).
%I	Acquisition des entrées	Ecriture en mémoire de l'état des informations des entrées TOR et spécifiques à l'application du module associées à la tâche.
-	Traitement du programme	Exécution du programme d'application écrit par l'utilisateur.
%Q	Mise à jour des sorties	Ecriture des bits ou des mots de sorties associés aux modules TOR et spécifiques à l'application, associés à la tâche selon l'état défini par le programmed'application.

Mode de fonctionnement

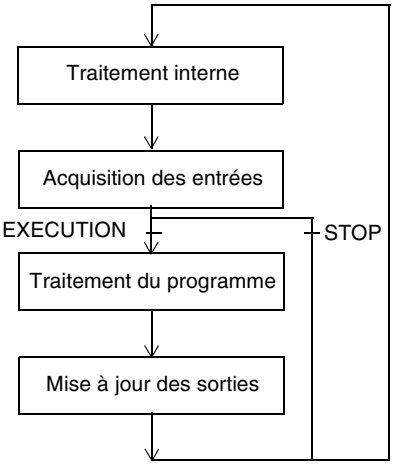
Automate en mode RUN, le processeur effectue les opérations suivantes :

- traitement interne
- acquisition des entrées
- traitement du programme d'application
- mise à jour des sorties

Automate en mode STOP, le processeur effectue les opérations suivantes :

- traitement interne
- acquisition des entrées

Illustration L'illustration suivante présente les cycles de fonctionnement.



Cycle de contrôle Le cycle de contrôle est effectué par le chien de garde.

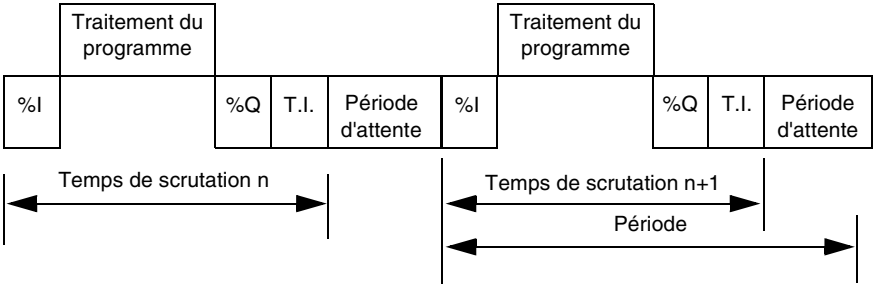
Scrutation périodique

Introduction

Dans ce mode de fonctionnement, l'acquisition des entrées, le traitement du programme d'application et la mise à jour des sorties s'effectuent de façon périodique selon un intervalle défini lors de la configuration (de 2 à 150 ms). Au début de la scrutation de l'automate, un temporisateur, dont la valeur est initialisée sur la période définie lors de la configuration, démarre le décomptage. La scrutation de l'automate doit se terminer avant la fin du décomptage et avant le début d'une nouvelle scrutation.

Fonctionnement

L'illustration suivante présente les phases d'exécution de la scrutation périodique.



Description des phases de fonctionnement

Le tableau suivant décrit les phases de fonctionnement.

Repère	Phase	Description
T.I.	Traitement interne	Le système réalise implicitement la surveillance de l'automate (gestion des bits et mots système, mise à jour des valeurs courantes de l'horodateur, mise à jour des voyants d'état, détection des commutateurs RUN/STOP, etc.) et le traitement des requêtes en provenance de TwidoSoft (modifications et animation).
%I	Acquisition des entrées	Ecriture en mémoire de l'état des informations des entrées TOR et spécifiques à l'application du module associées à la tâche.
-	Traitement du programme	Exécution du programme d'application écrit par l'utilisateur.
%Q	Mise à jour des sorties	Ecriture des bits ou des mots de sorties associés aux modules TOR et spécifiques à l'application, associés à la tâche selon l'état défini par le programme d'application.

Mode de fonctionnement

Automate en mode RUN, le processeur effectue les opérations suivantes :

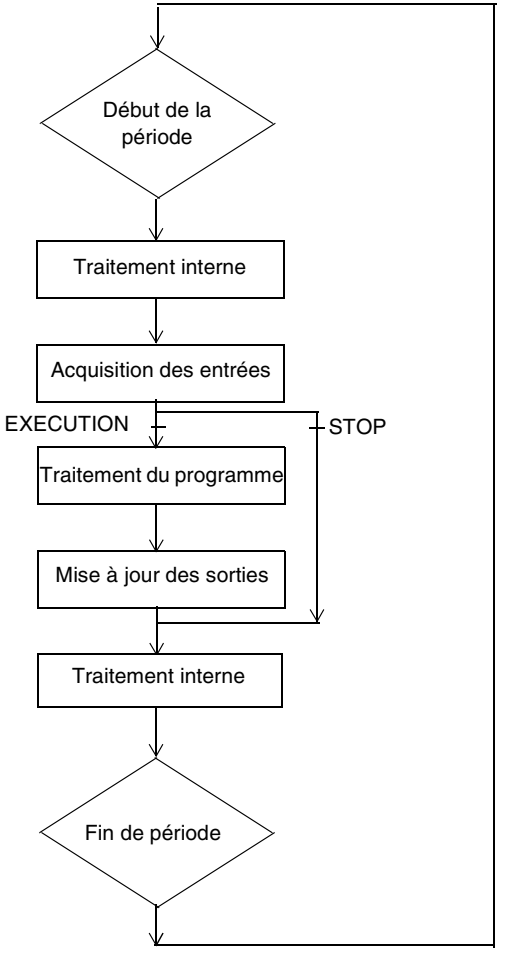
- ordre de traitement interne
- acquisition des entrées
- traitement du programme d'application
- mise à jour des sorties

Si la période n'est pas terminée, le processeur poursuit son cycle de fonctionnement jusqu'à la fin de la période du traitement interne. Si la durée de fonctionnement dépasse celle affectée à la période, l'automate signale un débordement de période en réglant le bit système %S19 de la tâche sur 1. Le traitement se poursuit jusqu'à son exécution totale. Néanmoins, il ne doit pas dépasser le temps limite du chien de garde. La scrutation suivante est enchaînée après l'écriture implicite des sorties de la scrutation en cours.

Automate en mode STOP, le processeur effectue les opérations suivantes :

- traitement interne
- acquisition des entrées

Illustration L'illustration suivante présente les cycles de fonctionnement.



Cycle de contrôle Deux contrôles sont effectués :

- débordement de période
- chien de garde

Vérification de la durée de scrutation

Généralités

Le cycle de la tâche maître est contrôlé par un temporisateur chien de garde appelé Tmax (durée maximale du cycle de la tâche maître). Ce temporisateur permet d'afficher les erreurs de l'application (boucles infinies, etc.) et garantit une durée maximale du rafraichissement des sorties.

Chien de garde logiciel (fonctionnement périodique ou cyclique)

Au cours du fonctionnement périodique ou cyclique, le déclenchement du chien de garde provoque une erreur logicielle. Cette application passe en mode HALT et règle le bit %S11 sur 1. La relance de la tâche nécessite une connexion à TwidoSoft afin d'analyser la cause de l'erreur, une modification de l'application pour corriger l'erreur, ainsi qu'une relance des requêtes INIT et RUN.

Note : L'état HALT correspond à l'arrêt immédiat de l'application causé par une erreur d'application logicielle telle qu'un débordement de scrutation. Les données retiennent les valeurs courantes, permettant ainsi l'analyse de la cause de l'erreur. Toutes les tâches de l'instruction actuelle sont arrêtées. La communication avec l'automate est disponible.

Contrôle en fonctionnement périodique

En fonctionnement périodique, un contrôle supplémentaire permet de détecter un dépassement de période :

- **%S19** indique que la période est dépassée. Il est réglé sur :
  - 1 par le système lorsque la durée de scrutation est supérieure à la durée de la tâche.
  - 0 par l'utilisateur.
- **%SW0** contient la valeur de la période (0-150 ms). Il est :
  - initialisé lors d'un démarrage à froid par la valeur réglée au moment de la configuration et
  - peut être modifié par l'utilisateur.

Exploitation des temps d'exécution de la tâche maître

Les mots système suivants permettent d'obtenir des informations sur le temps de cycle de l'automate :

- **%SW11** initialise la durée maximale du chien de garde (10 à 500 ms).
- **%SW30** contient le durée d'exécution du dernier cycle de scrutation de l'automate.
- **%SW31** contient la durée d'exécution du plus long cycle de scrutation de l'automate depuis le dernier démarrage à froid.
- **%SW32** contient la durée d'exécution du plus court cycle de scrutation de l'automate depuis le dernier démarrage à froid.

Note : Ces différentes informations sont également accessibles depuis l'éditeur de configuration.

Modes de fonctionnement

Introduction

Twido Soft est utilisé pour prendre en compte les trois groupes de modes de fonctionnement :

- vérification
- exécution ou production
- arrêt.

Note : Ces modes de fonctionnement sont définis dans l'ouvrage « Design Guide for Operating and Stopping Modes », publié par l'agence pour le développement des automatismes industriels (Applied Industrial Automation Development Agency).

Démarrage via Grafcet

Ces différents modes de fonctionnement sont accessibles depuis Grafcet ou en utilisant Grafcet, en appliquant les méthodes suivantes :

- initialisation de Grafcet
- préréglage des pas
- conservation d'une situation
- gel de diagrammes.

Le traitement préliminaire et l'utilisation de bits système garantissent une gestion efficace du mode de fonctionnement qui ne provoque aucune complication du programme utilisateur et qui n'implique aucune surcharge sur ce dernier.

Bits système Grafcet

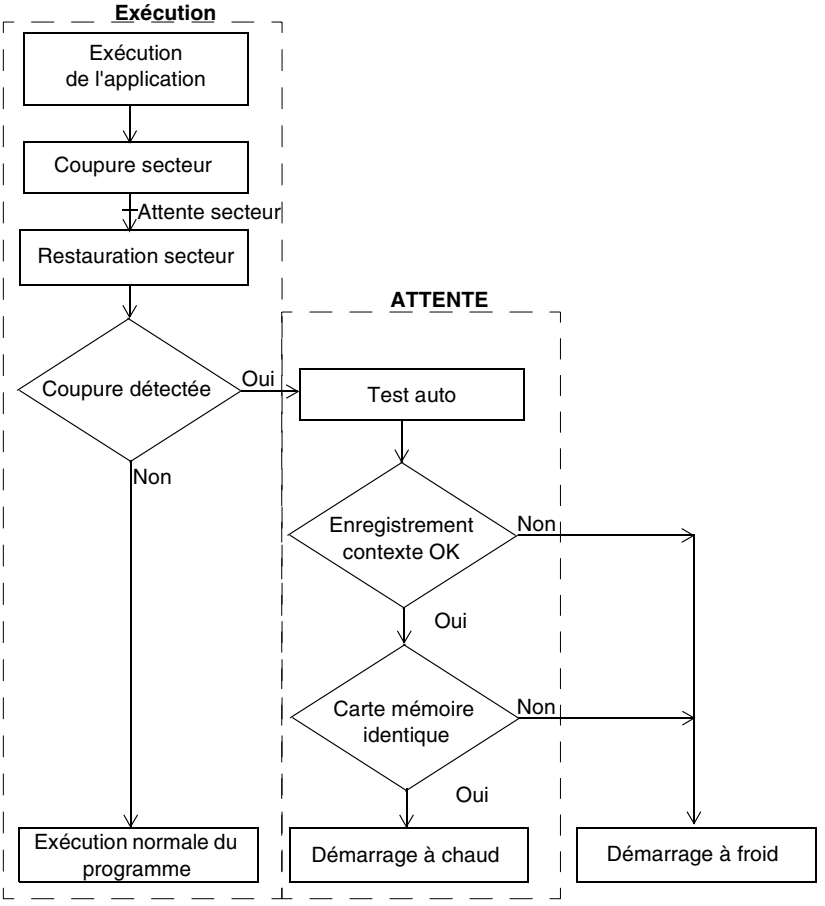
L'utilisation des bits %S21, %S22 et %S23 est réservée au traitement préliminaire. Ces bits sont automatiquement remis à zéro par le système, et ne doivent être écrits que par l'instruction Set **S**.
Le tableau suivant présente les bits système associés à Grafcet :

Bit	Fonction	Description
%S21	Initialisation de GRAFCET	Normalement réglé sur 0, ce bit est réglé sur 1 par : ● un démarrage à froid, **%S0=1** ; ● l'utilisateur, uniquement dans la section du programme de prétraitement, à l'aide de l'instruction Set **S %S21** ou d'une bobine Set **-(S)- %S21**. Conséquences: ● Désactivation de tous les pas. ● Activation de tous les pas initiaux.
%S22	GRAFCET RESET	Normalement réglé sur 0, ce bit peut être réglé sur 1, uniquement par le programme au cours du prétraitement. Conséquences : ● Désactivation de tous les pas. ● Arrêt de la scrutation du traitement séquentiel.
%S23	Préréglage et gel de GRAFCET	Normalement réglé sur 0, ce bit peut être réglé sur 1, uniquement par le programme au cours du prétraitement. ● Remise à zéro de Grafcet en réglant %S22 sur 1. ● Prépositionne les pas pour leur activation, par une série d'instructions S Xi. ● Activation du prépositionnement en réglant %S23 sur 1. Gel d'une situation : ● Dans la situation initiale : par le maintien de %S21 sur 1 par le programme. ● Dans une situation « vide » : par le maintien de %S22 sur 1 par le programme. ● Dans une situation déterminée par le maintien de %S23 sur 1.

Gestion des coupures et des reprises secteur

Illustration

L'illustration suivante présente les différentes reprises secteur détectées par le système. Si la durée de la coupure est inférieure au temps de filtrage de l'alimentation (environ 10 ms pour une alimentation en courant alternatif ou 1 ms pour une alimentation en courant continu), elle n'est pas prise en compte par le programme qui s'exécute normalement.



Note : Le contexte est enregistré dans une mémoire RAM sur batterie de secours. A la mise sous tension, le système vérifie l'état des batteries et du contexte enregistré afin de déterminer si un démarrage à chaud est possible.

Bit d'entrée Run/ Stop et option Démarrage automatique en Run

Le bit d'entrée Run/Stop est prioritaire sur l'option Démarrage automatique en Run accessible à partir de la boîte de dialogue Mode de scrutation (voir le guide d'exploitation TwidoSoft). Si le bit Run/Stop est défini, l'automate redémarre en mode Run à la reprise secteur.
Le mode de l'automate est déterminé de la façon suivante.

Bit d'entrée Run/Stop	Démarrage automatique en Run	Etat résultant
Zéro	Zéro	Arrêté
Zéro	Un	Arrêté
Front montant	Sans importance	En cours d'exécution
Un	Sans importance	En cours d'exécution
Non configuré dans le logiciel	Zéro	Arrêté
Non configuré dans le logiciel	Un	En cours d'exécution

Note : Pour tous les automates compacts, si l'automate est en mode Run à l'interruption du secteur et que l'indicateur "Démarrage automatique en Run" n'est pas sélectionné dans la boîte Mode de scrutation, l'automate redémarre en mode Stop à la reprise secteur.

Note : Pour tous les automates modulaires, si la batterie de l'automate fonctionne normalement lors de l'interruption du secteur, l'automate redémarre dans le mode effectif au moment de l'interruption. L'indicateur "Démarrage automatique en Run", sélectionné dans la boîte de dialogue Mode de scrutation, n'aura aucun effet sur le mode adopté à la reprise secteur.

Fonctionnement

Le tableau suivant décrit les phases du traitement des coupures secteur.

Phase	Description
1	Lors de la coupure secteur, le système mémorise le contexte application et l'heure de la coupure.
2	Il règle toutes les sorties dans un état de repli comme fonction des paramètres de sécurité (%S9).
3	A la reprise secteur, le contexte sauvegardé est comparé à celui en cours. Cette comparaison permet de définir le type de démarrage à exécuter : ● Si le contexte application a changé (perte du contexte système ou nouvelle application), l'automate effectue l'initialisation de l'application : démarrage à froid. ● Si le contexte application est identique, l'automate effectue une reprise sans initialisation des données : redémarrage à chaud.

Gestion d'une reprise à chaud

Cause d'une reprise à chaud

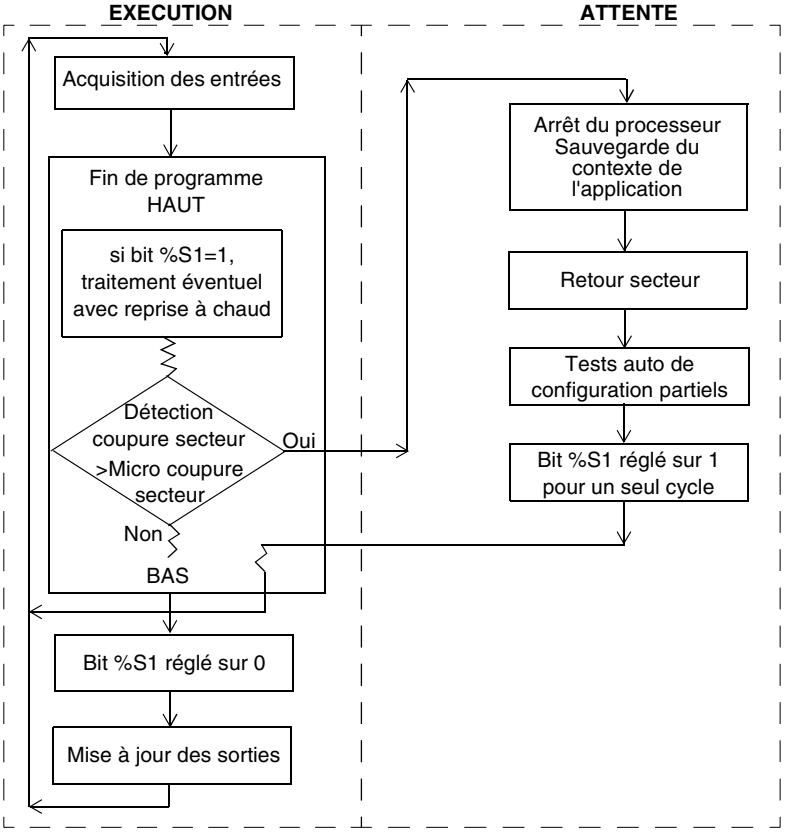
Une reprise à chaud peut être provoquée :

- par une reprise secteur sans perte du contexte,
- lorsque le bit système **%S1** est réglé sur 1 par le programme,
- depuis l'afficheur, lorsque l'automate est en mode STOP.

Note : Les automates compacts sont toujours mis sous tension en démarrage à froid. Les automates modulaires redémarrent toujours en reprise à chaud.

Illustration

Le dessin ci-après décrit le fonctionnement d'une reprise à chaud en mode d'exécution (RUN).



Reprise de l'exécution du programme

Le tableau suivant décrit les phases de reprise de l'exécution d'un programme après une reprise à chaud.

Phase	Description
1	L'exécution du programme reprend à partir de l'élément où a eu lieu la coupure secteur, sans mise à jour des sorties. **Remarque :** Seuls les éléments du code de l'utilisateur sont redémarrés. Le code système (la mise à jour des sorties, par exemple) n'est pas redémarré.
2	A la fin du cycle de reprise, le système : ● annule la réservation de l'application lorsqu'elle est réservée (et provoque une application STOP en cas de débogage) ; ● effectue la réinitialisation des messages.
3	Le système effectue un cycle de reprise au cours duquel il : ● relance la tâche avec les bits **%S1** (balise de reprise à chaud) et **%S13** (premier cycle en mode RUN) réglés sur 1, ● remet à l'état 0 les bits **%S1** et **%S13** à la fin de ce premier cycle de la tâche maître.

Gestion d'un démarrage à chaud

En cas de démarrage à chaud et lorsque le traitement d'une application particulière est requis, le bit **%S1** doit être testé en début du cycle de tâche et le programme correspondant doit être appelé.

Sorties après une coupure secteur

Dès qu'une coupure secteur est détectée, les sorties sont réglées sur un état de repli (par défaut) de 0.
A la reprise secteur, les sorties conservent leur dernier état jusqu'à ce qu'elles soient remises à jour par la tâche.

Gestion d'un démarrage à froid

Cause d'un démarrage à froid

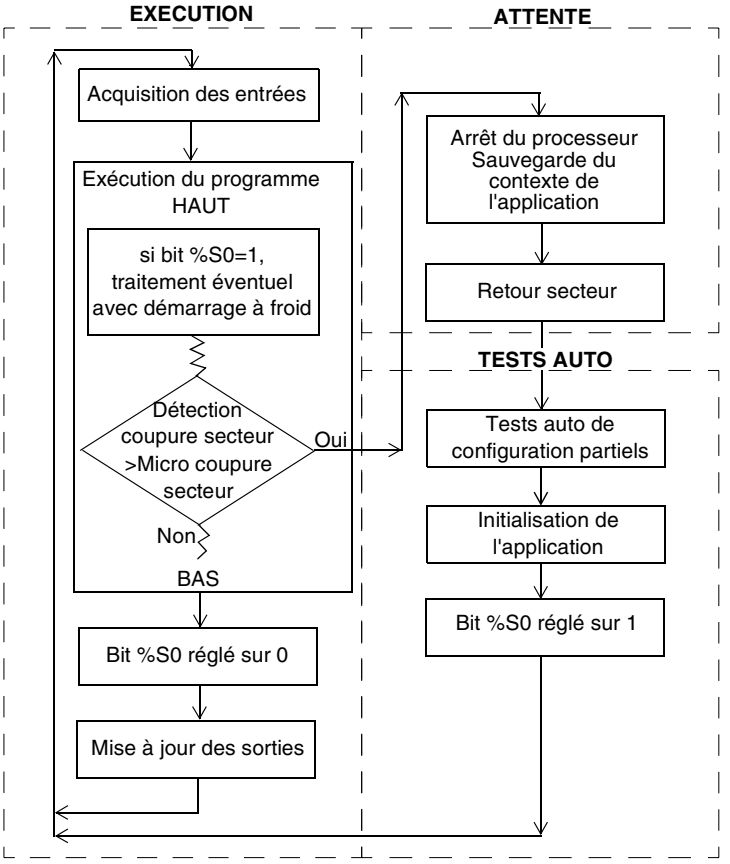
Un démarrage à froid peut être provoqué :

- par le chargement d'une nouvelle application dans la mémoire RAM,
- par une reprise secteur avec perte du contexte de l'application,
- lorsque le bit **%S0** est réglé sur 1 par le programme,
- depuis l'afficheur, lorsque l'automate est en mode STOP.

Note : Les automates compacts sont toujours mis sous tension en démarrage à froid. Les automates modulaires redémarrent toujours en reprise à chaud.

Illustration

Le dessin suivant décrit le fonctionnement d'une reprise à froid en mode d'exécution (RUN).



Fonctionnement

Le tableau ci-après décrit les phases de reprise de l'exécution du programme sur reprise à froid.

Phase	Description
1	A la mise sous tension, l'automate est en mode d'exécution (RUN). En cas de redémarrage à froid faisant suite à un arrêt causé par une ERREUR, le système impose une redémarrage à froid. L'exécution du programme reprend en début de cycle.
2	Le système effectue : ● une remise à 0 des bits et des mots internes et des images E/S ● l'initialisation des bits et mots système ; ● l'initialisation des blocs fonctions à partir des données de configuration.
3	Pour ce premier cycle de reprise, le système : ● relance la tâche avec les bits **%S0** (balise de reprise à froid) et **%S13** (premier cycle en mode RUN) réglés sur 1 ; ● remet à 0 les bits **%S0** et **%S13** à la fin de ce premier cycle de la tâche.

Gestion d'un démarrage à froid

Dans le cas d'un démarrage à froid et lorsque le traitement particulier d'une application est requis, le bit **%S0** (qui reste à 1) doit être testé au cours du premier cycle de la tâche.

Sorties après une coupure secteur

Dès qu'une coupure secteur est détectée, les sorties sont réglées sur un état de repli (par défaut) de 0.
A la reprise secteur, les sorties sont à zéro jusqu'à ce qu'elles soient remises à jour par la tâche.

Initialisation de l'automate

Introduction

Les automates peuvent être initialisés par TwidoSoft en réglant les bits système **%S0** (démarrage à froid) et **%S1** (reprise à chaud).

Initialisation en démarrage à froid

Pour une initialisation en démarrage à froid, le bit système **%S0** doit être réglé sur 1.

Initialisation en démarrage à chaud à l'aide de %S0 et de %S1

Pour une initialisation en démarrage à chaud, les bits système **%S1** et **%S0** doivent être réglés sur 1.
L'exemple suivant montre comment programmer une initialisation en reprise à chaud à l'aide des bits système.


```
LD  %S1   Si %S1 = 1 (reprise à chaud), le réglage de %S0 sur 1 initialise l'automate.
ST  %S0   Ces deux bits sont remis à zéro par le système à la fin de la
          scrutation suivante.
```

Note : Ne réglez pas %S0 sur 1 pour plus d'une scrutation de l'automate.

Initialisation en démarrage à chaud à l'aide de la commande INIT

Une initialisation en démarrage à chaud peut également être demandée à l'aide de la commande INIT. La commande INIT met l'automate à l'état IDLE et la réinitialisation des données de l'application et de l'état de la tâche à l'état STOPPED.

Fonctions spéciales

II

En bref...

Présentation

Cette rubrique décrit les communications, les fonctions analogiques intégrées et la gestion des modules d'E/S analogiques des automates Twido.

Contenu de cet intercalaire

Cet intercalaire contient les chapitres suivants :

Communications

5

En bref...

Présentation

Ce chapitre offre une présentation des procédures de configuration, de programmation et de gestion des communications à l'aide d'automates Twido.

Contenu de ce chapitre

Ce chapitre contient les sujets suivants :

Présentation des communications

Présentation

Twido dispose de deux ports série de communication utilisés pour communiquer avec les automates distants, les automates d'extension ou divers périphériques externes. Les deux ports, lorsqu'ils disponibles, peuvent être utilisés pour tous les services, à l'exception de la communication avec Twido Soft, qui ne peut se faire qu'avec le premier port. Trois différents protocoles de base sont pris en charge sur chaque automate Twido : liaison distante, ASCII ou modbus (maître modbus ou esclave modbus).

Liaison distante

La liaison distante est un bus maître/esclave très rapide conçu pour communiquer une petite quantité de données entre l'automate maître et un maximum de sept automates distants (esclave). Les données de l'application ou les données d'E/S sont transférées en fonction de la configuration des automates distants. Il est possible d'associer différents types d'automates, tels que des automates d'E/S distantes et des automates d'extension.

ASCII

Le protocole ASCII est un protocole semi-duplex en mode caractère simple utilisé pour transmettre et/ou recevoir une chaîne de caractères de ou vers un périphérique (imprimante ou terminal). Ce protocole est uniquement pris en charge via l'instruction « EXCH ».

Modbus

Le protocole modbus est un protocole maître/esclave qui permet à un maître et un seul d'obtenir des réponses provenant des esclaves ou d'agir sur requête. Le maître peut s'adresser aux esclaves individuellement ou envoyer un message de diffusion générale à tous les esclaves. Les esclaves renvoient un message (réponse) aux requêtes qui leur sont adressées individuellement. Les réponses aux requêtes de diffusion générale du maître ne sont pas renvoyées.

Maître modbus - Le mode maître modbus permet à l'automate Twido de commencer la transmission d'une requête modbus dont la réponse est attendue de la part de l'esclave. Le mode maître modbus est uniquement pris en charge via l'instruction « EXCH ». Les modes ASCII et RTU modbus sont pris en charge en mode maître modbus.

Esclave modbus - Le mode esclave modbus permet à l'automate Twido de répondre aux requêtes modbus d'un maître modbus. Il s'agit du mode de communication par défaut si aucune communication n'a été configurée. L'automate Twido prend en charge les données modbus standard, les fonctions de contrôle et les extensions de service pour l'accès aux objets. Les modes ASCII et RTU modbus sont pris en charge en mode esclave modbus.

Note : Il peut exister 32 noeuds maximum sur un réseau RS-485 (1 maître et jusqu' à 31 esclaves), dont les adresses peuvent être comprises entre 1 et 247.

Communications entre TwidoSoft et l'automate

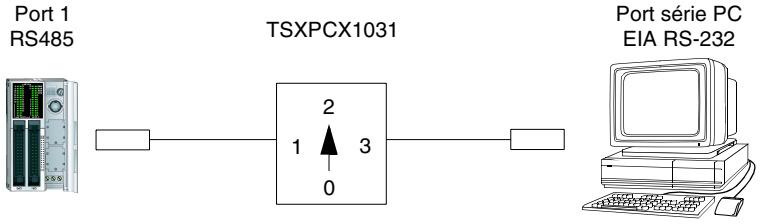
Présentation

Chaque automate Twido possède, sur son port 1, une prise de terminal EIA RS-485 intégrée. Cette prise possède sa propre alimentation interne. Faites transiter par le port 1 toutes les communications échangées avec le logiciel de programmation TwidoSoft. Aucune cartouche ou aucun module de communication optionnel(le) ne peut utiliser ce port.

	**ATTENTION** **ENDOMMAGEMENT DU MATERIEL INATTENDU** TwidoSoft risque de ne pas détecter de déconnexion lorsque vous retirez physiquement le câble de communication TSXPCX1031 d'un automate pour le réinsérer rapidement dans un autre automate. Afin d'éviter ce genre de problème, utilisez TwidoSoft pour effectuer la déconnexion avant de retirer le câble. **Le non-respect de ces précautions peut entraîner des lésions corporelles ou/et des dommages matériels.**

Raccordement du câble

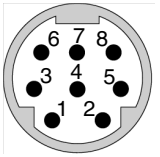
Le port EIA RS-232C de votre PC est raccordé au port 1 de l'automate à l'aide du câble de communication multifonctions TSXPCX1031. Ce câble, assurant la conversion des signaux entre EIA RS-232 et EIA RS-485, dispose d'un connecteur rotatif à 4 positions permettant de sélectionner les différents modes de fonctionnement. Les quatre positions de ce commutateur sont numérotées de 0 à 3. Pour les communications entre TwidoSoft et l'automate Twido, ce commutateur doit être positionné sur 2.
Ce raccordement est illustré dans le schéma suivant.



Note : Le signal DPT n'est pas mis à la terre. Le signal est réglé de manière interne afin d'indiquer au microprogramme de l'automate que la connexion courante est une connexion TwidoSoft.

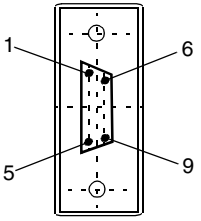
Brochages des connecteurs mâle et femelle

L'illustration suivante présente le brochage d'un connecteur mini DIN mâle à 8 broches.



Brochages	RS-485
1	A (+)
2	B (-)
3	NC
4	/DE
5	DPT
6	NC
7	0 V
8	5 V

L'illustration suivante présente le brochage d'un connecteur mini DIN femelle à 9 broches.



Brochages	RS-232
1	DCD
2	RX
3	TX
4	DTR
5	SG
6	NC
7	RTS
8	CTS
9	NC

Communications de liaison distante

Introduction

La liaison distante est un bus maître/esclave à haut débit conçu pour assurer l'échange d'une petite quantité de données entre l'automate maître et un maximum de sept automates (esclaves) distants. L'application ou les données d'E/S sont transférées, selon la configuration des automates distants. Il est possible d'associer différents types d'automates distants. Certains peuvent être des E/S distantes et d'autres des automates d'extension.

Note : L'automate maître contient les informations relatives au repère d'une E/S distante, mais il ne sait pas à quel automate précis correspond ce repère. Par conséquent, l'automate maître ne peut pas affirmer que toutes les entrées et sorties distantes utilisées dans l'application utilisateur existent réellement. Assurez-vous que cela soit le cas.

Note : Le bus d'E/S distantes et le protocole utilisé sont propriétaires et aucun périphérique tiers n'est autorisé sur le réseau.

ATTENTION

FONCTIONNEMENT DU MATERIEL INATTENDU

- Assurez-vous qu'il existe un seul automate maître sur une liaison distante et que chaque esclave dispose d'un repère unique. Le non-respect de cette précaution risque de corrompre les données ou de générer des résultats inattendus et ambigus.
- Assurez-vous que tous les esclaves disposent d'un repère unique. Deux esclaves ne doivent pas avoir le même repère. Le non-respect de cette précaution risque de corrompre les données ou de générer des résultats inattendus et ambigus.

Le non-respect de ces précautions peut entraîner des lésions corporelles ou/et des dommages matériels.

Note : La liaison distante nécessite une connexion EIA RS-485 et peut s'exécuter sur un seul port de communication à la fois.

Configuration matérielle

Une liaison distante doit utiliser un port EIA RS-485 à 3 fils minimum. Cela signifie qu'il est possible de la configurer afin d'utiliser le premier port ou un deuxième port optionnel existant.

Note : Un seul port de communication peut être configuré en tant que liaison distante.

Le tableau suivant répertorie les périphériques utilisables.

Périphérique	Port	Caractéristiques
TWDCAA10/16/24DRF, TWDLMDA20/40DUK, TWDLMDA20/40DTK, TWDLMDA20DRT	1	Base automate prenant en charge un port EIA RS-485 à 3 fils à l'aide d'un connecteur mini DIN.
TWDNOZ232D	2	Module de communication prenant en charge un port EIA RS-232 à 3 fils à l'aide d'un connecteur mini DIN. **Note :** Ce module n'est disponible que pour les automates modulaires. Lorsque le module est connecté, l'automate ne peut pas disposer d'un module d'expansion Afficheur.
TWDNOZ485D	2	Module de communication prenant en charge un port EIA RS-485 à 3 fils à l'aide d'un connecteur mini DIN. **Note :** Ce module n'est disponible que pour les automates modulaires. Lorsque le module est connecté, l'automate ne peut pas disposer d'un module d'expansion Afficheur.
TWDNOZ485T	2	Module de communication prenant en charge un port EIA RS-485 à 3 fils à l'aide d'un connecteur de borne. **Note :** Ce module n'est disponible que pour les automates modulaires. Lorsque le module est connecté, l'automate ne peut pas disposer d'un module d'expansion Afficheur.
TWDNAC232D	2	Adaptateur de communication prenant en charge un port EIA RS-232 à 3 fils à l'aide d'un connecteur mini DIN. **Note :** Cet adaptateur n'est disponible que pour les automates 16 et 24 E/S compacts et pour le module d'expansion Afficheur.
TWDNAC485D	2	Adaptateur de communication prenant en charge un port EIA RS-485 à 3 fils à l'aide d'un connecteur mini DIN. **Note :** Cet adaptateur n'est disponible que pour les automates 16 et 24 E/S compacts et pour le module d'expansion Afficheur.

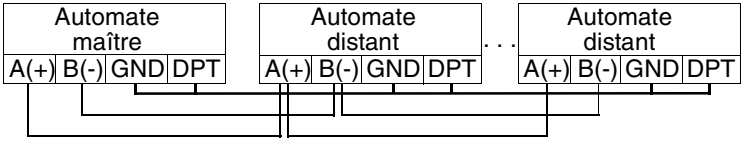
Périphérique	Port	Caractéristiques
TWDNAC485T	2	Adaptateur de communication prenant en charge un port EIA RS-485 à 3 fils à l'aide d'un connecteur de borne. **Note :** Cet adaptateur n'est disponible que pour les automates 16 et 24 E/S compacts et pour le module d'expansion Afficheur.
TWDXCPODM	2	Module d'expansion Afficheur prenant en charge un port EIA RS-232 à 3 fils à l'aide d'un connecteur mini DIN, un port EIA RS-485 à 3 fils à l'aide d'un connecteur mini DIN ou un port EIA RS-485 à 3 fils à l'aide d'un connecteur de borne. **Note :** Ce module n'est disponible que pour les automates modulaires. Lorsque le module est connecté, l'automate ne peut pas disposer d'un module d'expansion Communication.

Note : La configuration du port 2 (disponibilité et type) est uniquement contrôlée lors de la mise sous tension ou de la réinitialisation.

Connexion de câbles à chaque périphérique

Note : Le signal DPT sur la broche 5 doit être relié à la terre sur la broche 7 afin de signaler l'utilisation de communications de liaison distante. Lorsque ce signal n'est pas relié à la terre, l'automate Twido maître ou esclave est défini par défaut dans un mode dans lequel des tentatives d'établir des communications avec TwidoSoft s'effectuent.

Les connexions de câbles effectués à chaque périphérique sont représentées ci-dessous.



Note : La connexion DPT à GND (terre) n'est nécessaire qu'en cas de connexion à une base automate sur le port 1.

Configuration logicielle

Un seul automate maître doit être défini sur la liaison distante. En outre, chaque automate distant doit conserver un repère esclave unique. L'utilisation de repères identiques par plusieurs maîtres ou esclaves risque de corrompre des transmissions ou de créer des ambiguïtés.

	**ATTENTION**
	Mise en route non souhaitée d'équipements Assurez-vous qu'il existe un seul automate maître sur une liaison distante et que chaque esclave dispose d'un repère unique. Le non-respect de cette précaution risque de corrompre les données ou de générer des résultats inattendus et ambigus. **Le non-respect de ces précautions peut entraîner des lésions corporelles ou/et des dommages matériels.**

Configuration de l'automate maître

Configurez l'automate maître à l'aide de TwidoSoft pour gérer un réseau de liaison distante constitué au maximum de sept automates distants. Le maître prend en charge un mélange hétérogène d'automates distants (soit des E/S distantes, soit des automates d'extension) sur la liaison distante. Le repère du maître configuré à l'aide de TwidoSoft correspond au repère 0.

Configuration de l'automate distant

Vous pouvez utiliser chacun des automates distants en tant qu'E/S distantes ou automate d'extension. Ceux-ci sont configurés à l'aide de TwidoSoft afin de leur affecter un repère compris entre 1 et 7 (notez que 0 est réservé au maître de la liaison distante).
Le tableau suivant résume les différences et les contraintes de chacun de ces types de configuration d'automate distant.

Type	Programme d'application	Accès aux données
E/S distantes	Non Pas même une simple instruction "END"	%I et %Q Seule l'E/S locale de l'automate distant est accessible (et non son E/S d'expansion).
Automate d'extension	Oui Le mode Run n'est pas connecté à celui du maître.	%INW et %QNW Il est possible de transmettre un maximum de quatre mots d'entrée et quatre mots de sortie vers et depuis chaque extension.

Synchronisation de scrutation de l'automate distant

Le cycle de mise à jour de la liaison distante n'est pas synchronisé avec la scrutation de l'automate maître. Les communications avec les automates distants sont déclenchées par interruption et se produisent en tant que tâches en arrière-plan, en parallèle avec l'exécution de la scrutation de l'automate maître. A la fin du cycle de scrutation, les valeurs les plus récentes sont lues dans les données d'application à utiliser pour la prochaine solution. Ce traitement est le même pour les automates d'E/S distantes et d'extension.

Tous les automates peuvent vérifier l'activité de la liaison générale à l'aide du bit système %S111. Mais pour accomplir la synchronisation, un automate maître ou d'extension doit utiliser le bit système %S110. Le réglage est effectué sur 1 une fois qu'un cycle de mise à jour complet s'est déroulé. Le programme d'application est responsable de sa remise à 0.

Le maître peut activer ou désactiver la liaison distante à l'aide du bit système %S112. Les automates peuvent contrôler la configuration et l'état de santé de la liaison distante à l'aide de %S113. Le signal DPT sur le port 1 (utilisé pour déterminer si TwidoSoft est connecté) est détecté et signalé sur %S100.

Le tableau suivant résume toutes ces informations.

Bit système	Etat	Indication
%S100	0	maître/esclave : DPT inactif (câble TwidoSoft NON connecté)
	1	maître/esclave : DPT actif (câble TwidoSoft connecté)
%S110	0	maître/esclave : réinitialisé par l'application
	1	maître : tous les échanges de liaison distante effectués (E/S distantes uniquement) esclave : échange avec maître effectué
%S111	0	maître : échange de liaison distante unique effectué esclave : échange de liaison distante unique détecté
	1	maître : échange de liaison distante unique actif esclave : échange de liaison distante unique détecté
%S112	0	maître : liaison distante désactivée
	1	maître : liaison distante activée
%S113	0	maître/esclave : configuration/fonctionnement de la liaison distante OK
	1	maître : erreur de configuration/fonctionnement de la liaison distante esclave : erreur de fonctionnement de la liaison distante

Redémarrage de l'automate maître

Lorsqu'un automate maître redémarre, l'un des événements suivants se produit :

- Un démarrage à froid (%S0 = 1) force la réinitialisation des communications.
- Un démarrage à chaud (%S1 = 1) force la réinitialisation des communications.
- En mode Stop, le maître continue à communiquer avec les esclaves, le bit Run/Stop étant réglé afin d'indiquer Stop.

Redémarrage de l'automate esclave

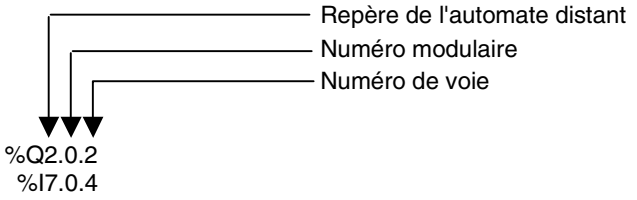
Lorsqu'un automate esclave redémarre, l'un des événements suivants se produit :

- Un démarrage à froid (%S0 = 1) force la réinitialisation des communications.
- Un démarrage à chaud (%S1 = 1) force la réinitialisation des communications.
- En mode Stop, l'esclave continue de communiquer avec le maître. Si le maître indique qu'une requête Stop :
  - Les E/S distantes affectent une interruption.
  - L'automate d'extension continue dans son état actuel.

Accès aux données E/S distantes

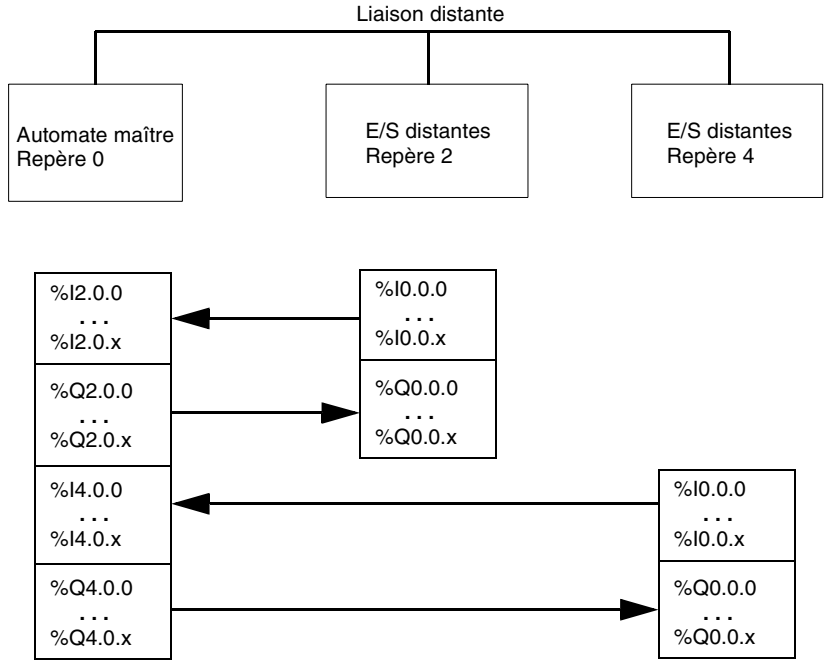
L'automate distant configuré en tant que E/S distantes ne possède, ni n'exécute son propre programme d'application. Les entrées et sorties TOR de base de l'automate distant sont une simple extension de celles de l'automate maître. L'application doit uniquement utiliser le mécanisme de repérage complet à trois chiffres fourni.

Note : Le numéro de module est toujours zéro pour les E/S distantes.



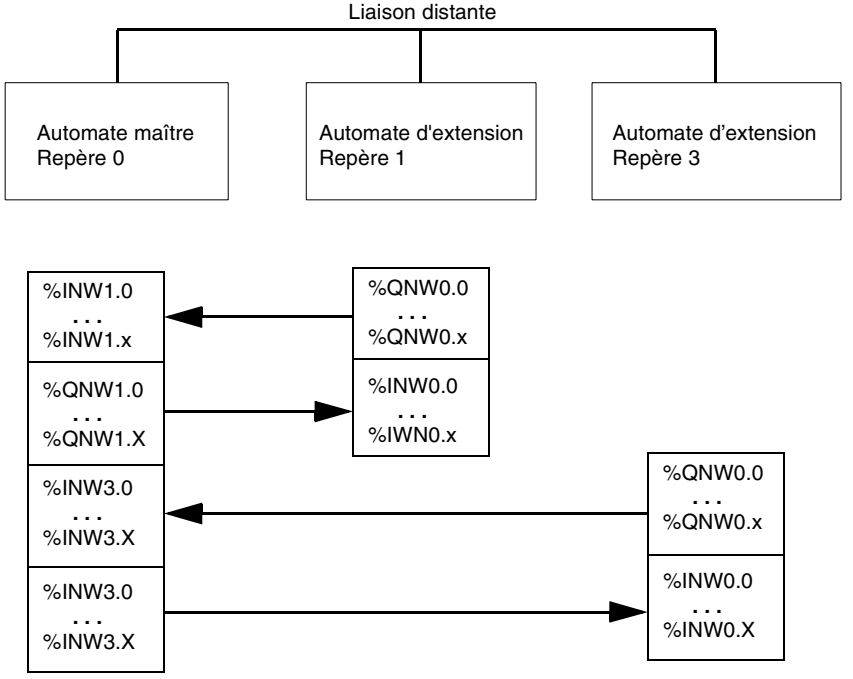
Pour communiquer avec les E/S distantes, l'automate maître utilise la notation d'entrée et sortie standard %I et %Q. Pour accéder au troisième bit de sortie de l'E/S distante configurée au repère 2, le maître doit définir %Q2.0.2. De même, pour lire le cinquième bit d'entrée des E/S distantes configurée à l'emplacement 7, le maître doit charger %I7.0.4.

Note : L'accès du maître est restreint aux E/S TOR appartenant aux E/S locales de l'automate distant. Aucune E/S analogique ou d'expansion ne peut être transférée, hormis en cas d'utilisation de communications d'extension.


Liaison distante
Automate maître
Repère 0
E/S distantes
Repère 2
E/S distantes
Repère 4
%I2.0.0
. . .
%I2.0.x
%Q2.0.0
. . .
%Q2.0.x
%I4.0.0
. . .
%I4.0.x
%Q4.0.0
. . .
%Q4.0.x
%I0.0.0
. . .
%I0.0.x
%Q0.0.0
. . .
%Q0.0.x
%I0.0.0
. . .
%I0.0.x
%Q0.0.0
. . .
%Q0.0.x

Accès aux données de l'automate d'extension

Pour communiquer avec des automates d'extension, le maître utilise des mots réseau %INW et %QNW afin d'échanger des données. Chaque extension du réseau est accessible par son repère distant "j" à l'aide de mots %INWj.k et %QNWj.k. Chaque automate d'extension du réseau utilise %INW0.0 à %INW0.3 et %QNW0.0 à %QNW0.3 pour accéder aux données situées sur le maître. Les mots réseau sont automatiquement mis à jour lorsque les automates sont en mode RUN ou STOPPED.
L'exemple suivant illustre l'échange d'un maître avec deux automates d'extension configurés.



Il n'existe aucune remise de messages d'égal à égal au sein de la liaison distante. Il est possible d'utiliser des programmes d'application de concert avec des mots réseau, afin de transférer des informations entre des automates distants, qui utilisent effectivement le maître en tant que pont.

Informations d'état

Outre les bits système décrits précédemment, le maître conserve l'état de présence et de configuration des automates distants. Cette action s'effectue dans les mots systèmes %SW111 et %SW113. L'automate maître ou l'automate distant peuvent obtenir la valeur de la dernière erreur survenue pendant la communication sur la liaison distante dans le mot système %SW112.
Chacune de ces erreurs est détaillée dans le tableau suivant.

Mots système	Utilisation	
%SW111		Etat de la liaison distante : deux bits pour chaque automate distant (maître uniquement)
	x0-5	0 - automate distant 1-6 absent
		1 - automate distant 1-6 présent
	x6	0 - automate distant 7 absent
		1 - automate distant 7 présent
	x8-13	0 - E/S distantes détectées sur l'automate distant 1-6
		1 - automate d'extension détecté sur l'automate distant 1-6
	x14	0 - E/S distantes détectées sur l'automate distant 7
		1 - automate d'extension détecté sur l'automate distant 7
%SW112		Code d'erreur de configuration ou de fonctionnement de la liaison distante :
		0 - opérations réussies
		1 - expiration du délai (esclave)
		2 - erreur de checksum détectée (esclave)
		3 - incohérence de configuration (esclave)
%SW113		Configuration de la liaison distante : deux bits pour chaque automate distant (maître uniquement)
	x0-5	0 - automate distant 1-6 non configuré
		1 - automate distant 1-6 configuré
	x6	0 - automate distant 7 non configuré
		1 - automate distant 7 configuré
	x8-13	0 - E/S distantes configurées en tant qu'automate distant 1-6
		1 - automate d'extension configuré en tant qu'automate distant 1-6
	x14	0 - E/S distantes configurées en tant qu'automate distant 7
		1 - automate d'extension configuré en tant qu'automate distant 7

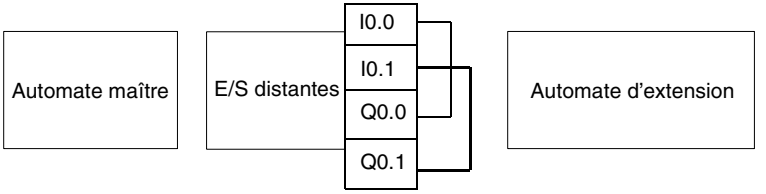
Exemple de liaison distante

Pour configurer une liaison distante, vous devez procéder comme suit :

1. Configurez le matériel.
2. Connectez le câblage de l'automate.
3. Connectez le câble de communication entre le PC et les automates.
4. Configurez le logiciel.
5. Ecrivez une application.

Les illustrations suivantes représentent une utilisation de la liaison distante avec les E/S distantes et un automate d'extension.

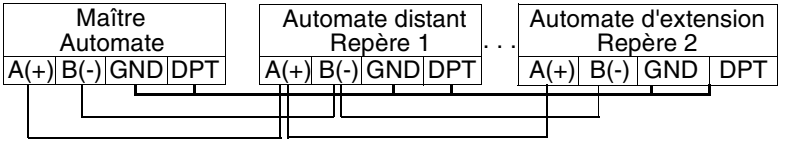
Etape 1 : Configuration du matériel



La configuration matérielle comprend trois bases automates de tout type. Le port 1 est utilisé en mode double. L'un des modes permet de configurer et de transférer le programme d'application à l'aide de TwidoSoft. Le second mode est destiné au réseau de liaison distante. Si un port 2 optionnel est disponible sur l'un des automates, il est possible de l'utiliser, mais tout automate prend en charge une seule liaison distante.

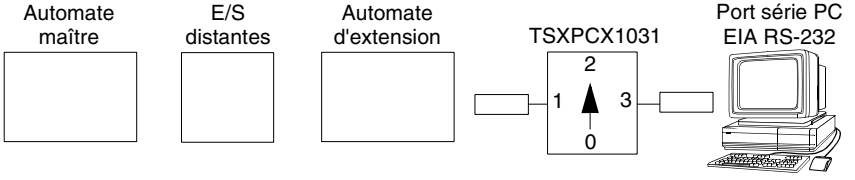
Note : Dans cet exemple, les deux premières entrées sur les E/S distantes sont câblées sur ses sorties.

Etape 2 : Connexion du câblage de l'automate



Connectez les fils des signaux D(+) et D(-) ensemble. Sur chaque automate, le signal DPT est relié à la terre. Bien que la mise à la terre du signal ne soit pas obligatoire pour une utilisation avec une liaison distante sur le port 2 (cartouche ou module de communication optionnels), il s'agit d'une bonne habitude à prendre.

Etape 3 : Connexion du câble de communication entre le PC et les automates



Le câble de programmation multifonctions TSXPCX1031 est utilisé pour communiquer avec chacune des trois bases automates. Assurez-vous que le commutateur du câble est en position 2. Afin de programmer chaque automate, il est nécessaire d'établir une communication point à point avec chaque automate. Pour établir cette communication : connectez-vous au port 1 du premier automate, transférez la configuration et les données de l'application, puis définissez l'automate sur l'état Run. Répétez cette procédure pour chaque automate.

Note : Il est nécessaire de déplacer le câble après chaque configuration d'automate et transfert d'application.

Une fois que les trois automates sont programmés, connectez les automates au réseau de liaison distante, comme indiqué à l'étape 2.
Etape 4 : Configuration du logiciel

Réglage comm. de l'automate	**Réglage comm. de l'automate**	**Réglage comm. de l'automate**
Type : Liaison distante Repère : 0 (maître)	Type : Liaison distante Repère : 1	Type : Liaison distante Repère : 2

Ajouter automates distants
Utilisation automate : E/S distantes Repère distant : 1 Utilisation automate : extension Repère distant : 2

Chacun des trois automates utilise TwidoSoft pour créer une configuration, et le cas échéant, le programme d'application. Pour l'automate maître, éditez la configuration de la communication de l'automate afin de régler le protocole sur "Liaison distante" et le repère sur "0 (Maître)".

Note : Vous pouvez configurer un seul automate en tant que maître sur une liaison distante.

Dans TwidoSoft, ajoutez une "E/S distante" au repère "1" et un "Automate d'extension" au repère "2".
Pour l'automate configuré en tant qu'E/S distantes, vérifiez que la configuration de la communication de l'automate est réglée sur "Liaison distante" et le repère sur "1".
Pour l'automate configuré en tant qu'extension, vérifiez que la configuration de la communication de l'automate est réglée sur "Liaison distante" et le repère sur "2".

Etape 5 : Ecriture d'une application

```
LD 1

[%MW0 := %MW0 +1]
[%QNW2.0 := %MW0]
[%MW1 := %INW2.0]

LD %I0.0
ST %Q1.0.0
LD %I1.0.0
ST %Q0.0

LD %I0.1
ST %Q1.0.1
LD %I1.0.1
ST %Q0.1
```

```
LD 1
[%QNW0.0 := %INW0.0]
```

Dans cet exemple, l'application maître incrémente un mot mémoire interne et le communique à l'automate d'extension à l'aide d'un seul mot réseau. L'automate d'extension prend le mot reçu du maître et le renvoie. Dans le maître, un mot mémoire différent reçoit et stocke cette transmission.
Pour communiquer avec l'automate d'E/S distantes, le maître envoie ses entrées locales aux sorties des E/S distantes. A l'aide de la connexion E/S externe des E/S distantes, les signaux sont renvoyés et récupérés par le maître.

Note : Cette communication se produit sous l'application du maître. Il n'existe aucune application dans l'automate d'E/S distantes.

Communications ASCII

Introduction

Le protocole ASCII offre aux automates Twido un protocole de mode caractère semi-duplex simple permettant de transférer et/ou de recevoir des données à l'aide d'un seul périphérique. Ce protocole est pris en charge à l'aide de l'instruction EXCHx et régulé à l'aide du bloc fonction %MSGx.

Les trois types de communications suivants sont possibles à l'aide du protocole ASCII :

- Transmission seule
- Transmission/réception
- Réception seule

La taille maximale des trames transmises et/ou reçues à l'aide de l'instruction EXCHx s'élève à 128 octets.

Configuration matérielle

Il est possible d'établir une liaison ASCII sur le port EIA RS-232 ou EIA RS-485 et de l'exécuter simultanément sur deux ports de communication au maximum. Le tableau suivant répertorie les périphériques utilisables.

Périphérique	Port	Caractéristiques
TWDCAA10/16/24DRF, TWDLMDA20/40DUK, TWDLMDA20/40DTK, TWDLMDA20DRT	1	Base automate prenant en charge un port EIA RS-485 à 3 fils à l'aide d'un connecteur mini DIN.
TWDNOZ232D	2	Module de communication prenant en charge un port EIA RS-232 à 3 fils à l'aide d'un connecteur mini DIN. **Note :** Ce module n'est disponible que pour les automates modulaires. Lorsque le module est connecté, l'automate ne peut pas disposer d'un module d'expansion Afficheur.
TWDNOZ485D	2	Module de communication prenant en charge un port EIA RS-485 à 3 fils à l'aide d'un connecteur mini DIN. **Note :** Ce module n'est disponible que pour les automates modulaires. Lorsque le module est connecté, l'automate ne peut pas disposer d'un module d'expansion Afficheur.
TWDNOZ485T	2	Module de communication prenant en charge un port EIA RS-485 à 3 fils à l'aide d'un connecteur de borne. **Note :** Ce module n'est disponible que pour les automates modulaires. Lorsque le module est connecté, l'automate ne peut pas disposer d'un module d'expansion Afficheur.
TWDNAC232D	2	Adaptateur de communication prenant en charge un port EIA RS-232 à 3 fils à l'aide d'un connecteur mini DIN. **Note :** Cet adaptateur n'est disponible que pour les automates 16 et 24 E/S compacts et pour le module d'expansion Afficheur.
TWDNAC485D	2	Adaptateur de communication prenant en charge un port EIA RS-485 à 3 fils à l'aide d'un connecteur mini DIN. **Note :** Cet adaptateur n'est disponible que pour les automates 16 et 24 E/S compacts et pour le module d'expansion Afficheur.
TWDNAC485T	2	Adaptateur de communication prenant en charge un port EIA RS-485 à 3 fils à l'aide d'un connecteur de borne. **Note :** Cet adaptateur n'est disponible que pour les automates 16 et 24 E/S compacts et pour le module d'expansion Afficheur.

Périphérique	Port	Caractéristiques
TWDXCPODM	2	Module d'expansion Afficheur prenant en charge un port EIA RS-232 à 3 fils à l'aide d'un connecteur mini DIN, un port EIA RS-485 à 3 fils à l'aide d'un connecteur mini DIN ou un port EIA RS-485 à 3 fils à l'aide d'un connecteur de borne. **Note :** Ce module n'est disponible que pour les automates modulaires. Lorsque le module est connecté, l'automate ne peut pas disposer d'un module d'expansion Communication.

Note : La configuration du port 2 (disponibilité et type) est uniquement contrôlée lors de la mise sous tension ou de la réinitialisation par le microprogramme de l'automate.

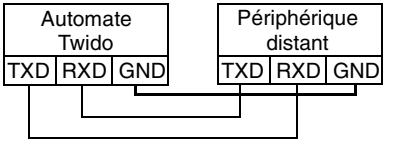
Câblage nominal

Les connexions de câble nominal sont représentées ci-dessous pour les types EIA RS-232 et EIA RS-485.

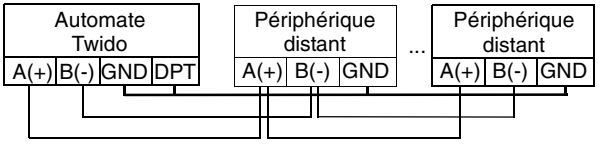
Note : Si vous utilisez le port 1 de l'automate Twido, vous devez connecter le signal DPT à GND (terre). Ce signal permet d'indiquer à l'automate Twido que les communications via le port 1 relèvent du protocole ASCII et non du protocole utilisé pour communiquer avec le logiciel TwidoSoft. Ce périphérique distant spécifique peut requérir l'utilisation de signaux supplémentaires (DTR, DSR, etc.).

Les connexions de câbles effectués à chaque périphérique sont représentées ci-dessous.

Câble EIA RS-232



Câble EIA RS-485



Note : La connexion DPT à GND (terre) n'est nécessaire qu'en cas de connexion à une base automate sur le port 1.

Configuration logicielle

Pour configurer l'automate afin d'utiliser une connexion en série pour envoyer et recevoir des caractères à l'aide du protocole ASCII, vous devez procéder comme suit.

Etape	Description
1	Configurez le port série pour le protocole ASCII à l'aide de TwidoSoft.
2	Créez dans votre application un tampon de transmission/réception pour que le protocole ASCII utilise l'instruction EXCHx.

Configuration du port

Un automate Twido peut utiliser son port 1 principal ou un port 2 configuré en option pour utiliser le protocole ASCII. Pour configurer un port série pour le protocole ASCII :

Etape	Action
1	Définissez tous les modules ou cartouches optionnels supplémentaires physiquement configurés sur la base.
2	Cliquez avec le bouton droit sur le port, puis cliquez sur Editer le paramétrage communication de l'automate... et modifiez le type du port série par "ASCII".
3	Définissez les paramètres de communication associés.

Configuration du tampon de transmission/ réception pour ASCII

La taille maximale des trames transmises et/ou reçues s'élève à 128 octets. En outre, la table de mots associée à l'instruction EXCHx se compose à la fois des tables de transmission et de réception.

	Octet le plus significatif	Octet le moins significatif
Mots de commande	Commande	Longueur (Transmetteur/ Récepteur)
	Réservés (0)	Réservés (0)
Table de transmission	Octet 1 transmis	Octet 2 transmis
	...	...
	...	Octet n transmis
	Octet n+1 transmis	
Table de réception	Octet 1 reçu	Octet 2 reçu
	...	...
	...	Octet p reçu
	Octet p+1 reçu	

Paramètres de contrôle

L'octet **Longueur** comprend la longueur à transmettre, qui est écrasée par le nombre de caractères reçus à la fin de la réception, si la réception est requise. L'octet **Commande** doit contenir l'un des éléments suivants :

- 0: Transmission seule
- 1: Transmission/réception
- 2: Réception seule

Tables de transmission/ réception

En mode Transmission seule, les tables de contrôle et de transmission sont renseignées avant l'exécution de l'instruction EXCHx ; elles peuvent être de type %KW ou %MW. Aucun espace n'est requis pour la réception des caractères en mode Transmission seule. Une fois que tous les octets ont été transmis, l'état de %MSGx.D est réglé sur 1 ; il est alors possible d'exécuter une nouvelle instruction EXCHx.
En mode Transmission/Réception, les tables de contrôle et de transmission sont renseignées avant l'exécution de l'instruction EXCHx ; elles doivent être de type %MW. Un espace prévu pour un maximum de 128 octets de réception est requis à la fin de la table de transmission. Une fois que tous les octets ont été transmis, l'automate Twido passe en mode de réception et est prêt à recevoir des octets.
En mode Réception seule, la table de contrôle est renseignée avant l'exécution de l'instruction EXCHx ; elle doit être de type %MW. Un espace prévu pour un maximum de 128 octets de réception est requis à la fin de la table de contrôle. L'automate Twido passe immédiatement en mode de réception et est prêt à recevoir des octets.
La réception est terminée une fois que l'octet de fin de trame a été reçu ou lorsque la table de réception est pleine. Si un délai différent de zéro est configuré, la réception se termine lorsque ce délai est écoulé. Si vous sélectionnez une valeur de délai égale à zéro, il n'existe aucun délai de réception. Par conséquent, pour arrêter la réception, vous devez activer l'entrée %MSGx.R.
Il n'existe aucun repérage inhérent associé au protocole ASCII, sauf si le périphérique unique dispose d'un repérage intégré dans le protocole ; toutefois l'automate Twido ne le prend pas en charge.

Echange de messages

Il est possible de configurer l'automate Twido pour envoyer et/ou recevoir des messages en mode caractère. Le langage propose deux services pour celui-ci :

- **Instruction EXCHx :** pour transmettre/recevoir des messages,
- **Bloc fonction %MSGx :** pour contrôler les échanges de messages.

L'automate Twido utilise le protocole configuré pour ce port lors du traitement d'une instruction EXCHx.

Note : Il est possible de configurer chaque port de communication pour différents protocoles ou pour le même protocole. Pour accéder à l'instruction EXCHx ou au bloc fonction %MSGx de chaque port de communication, il suffit d'y ajouter le numéro du port (1 ou 2).

Instruction EXCHx

L'instruction EXCHx permet à l'automate Twido d'envoyer et/ou de recevoir des informations vers/depuis des périphériques ASCII. L'utilisateur définit une table de mots (%MWi:L ou %KWi:L) contenant des informations de contrôle ainsi que les données à envoyer et/ou à recevoir (jusqu'à 64 mots de données dans la transmission et/ou réception). La description du format de la table de mots a été donnée précédemment.

Un échange de messages s'effectue à l'aide de l'instruction EXCHx.

Syntaxe : [EXCHx %MWi:L] ou [EXCHx %KWi:L]

où : x = numéro du port (1 ou 2)

L = nombre de mots du tableau des mots

L'automate Twido doit terminer l'échange de la première instruction EXCHx avant de pouvoir en lancer une deuxième. Il est nécessaire d'utiliser le bloc fonction %MSGx lors de l'envoi de plusieurs messages.

Le traitement de l'instruction par liste EXCHx se produit immédiatement, en sachant que toutes les transmissions sont démarrées sous contrôle d'interruptions (la réception des données est également sous contrôle d'interruptions), ce qui est considéré comme un traitement en arrière-plan.

Bloc fonction %MSGx

L'utilisation du bloc fonction %MSGx est facultative ; il permet de gérer des échanges de données. Le bloc fonction %MSGx remplit trois fonctions :

- **Vérification des erreurs de communications**
  La recherche d'erreurs permet de vérifier que la longueur du bloc (table de mots) programmée à l'aide de l'instruction EXCHx est suffisamment grande pour contenir la longueur du message à envoyer. Celle-ci est comparée à la longueur programmée dans l'octet le moins significatif du premier mot de la table de mots.
- **Coordination de plusieurs messages**
  Pour garantir la coordination lors de l'envoi de plusieurs messages, le bloc fonction %MSGx fournit les informations requises pour déterminer le moment où la transmission du message précédent est terminée.
- **Transmission de messages prioritaires**
  Le bloc fonction %MSGx vous permet de suspendre la transmission d'un message afin d'envoyer un message plus urgent.

Le bloc fonction %MSGx dispose d'une entrée et de deux sorties qui lui sont associées :

Entrée/Sortie	Définition	Description
R	Entrée RAZ	Réglée sur 1 : réinitialise la communication ou le bloc (%MSGx.E = 0 et %MSGx.D = 1).
%MSGx.D	Communication terminée	0: requête en cours. 1: communication terminée en cas de fin de transmission, de réception du caractère de fin, d'erreur ou de réinitialisation du bloc.
%MSGx.E	erreur	0: longueur du message OK et liaison OK. 1: en cas de mauvaise commande, de table configurée de manière incorrecte, de mauvais caractère reçu (débit, parité, etc.) ou de saturation de la table de réception.

Limitations

Il est important de garder à l'esprit les limitations suivantes :

- La disponibilité et le type du port 2 sont uniquement contrôlés lors de la mise sous tension ou de la réinitialisation.
- Tout message en cours de traitement sur le port 1 est abandonné lorsque TwidoSoft est connecté.
- Il est impossible de traiter EXCHx ou %MSG sur un port configuré en tant que liaison distante.
- EXCHx abandonne le traitement Modbus esclave actif (à l'exception du traitement TwidoSoft).
- Le traitement des instructions EXCHx ne fait pas l'objet d'une nouvelle tentative en cas d'erreur.
- Il est possible d'utiliser R %MSGx pour annuler le traitement de la réception d'une instruction EXCHx.
- Il est possible de configurer des instructions EXCHx avec un délai d'annulation de réception.
- Les messages multiples sont contrôlés via %MSGx.D.

Conditions d'erreur et de mode de fonctionnement

Si une erreur se produit lors de l'utilisation de l'instruction EXCHx, les bits %MSGx.D et %MSGx.E sont réglés sur 1, le mot système %SW63 contient le code d'erreur du port 1 et %SW64 le code d'erreur du port 2.

Mots système	Utilisation
%SW63	Code d'erreur EXCH1 : 0 - opération réussie 1 - tampon de transmission trop important (> 128) 2 - tampon de transmission trop petit 3 - table de mots trop petite 4 - débordement de la table de réception 5 - délai écoulé 6 - erreur de transmission (erreur reçue en réponse) 7 - mauvaise commande dans la table 8 - port sélectionné non configuré/disponible 9 - erreur de réception 10 - impossible d'utiliser %KW en cas de réception 11 - décalage de transmission plus important que la table de transmission 12 - décalage de réception plus important que la table de réception 13 - interruption du traitement EXCH par l'automate
%SW64	Code d'erreur EXCH2 : Voir %SW63.

Redémarrage de l'automate maître/esclave

Lorsqu'un automate maître/esclave redémarre, l'un des événements suivants se produit :

- Un démarrage à froid (%S0 = 1) force la réinitialisation des communications.
- Un démarrage à chaud (%S1 = 1) force la réinitialisation des communications.
- En mode Stop, l'automate arrête toutes les communications ASCII.

Exemple de liaison ASCII

Pour configurer une liaison ASCII, vous devez procéder comme suit :

1. Configurez le matériel.
2. Connectez le câble de communication ASCII.
3. Configurez le port.
4. Ecrivez une application.
5. Initialisez l'éditeur de tables d'animation.

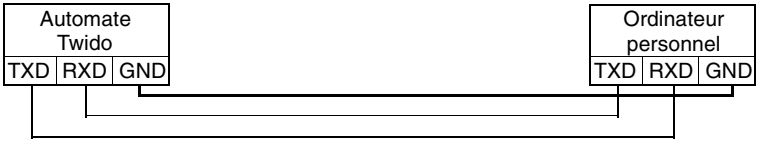
L'illustration suivante représente l'utilisation des communications ASCII à l'aide d'un émulateur de terminal sur un PC.

Etape 1 : Configuration du matériel

La configuration matérielle comporte deux connexions en série entre le PC et un automate Twido doté d'un port 2 EIA RS-232 optionnel. Sur un automate modulaire, le port 2 optionnel correspond à TWDNOZ232D. Sur l'automate compact, le port 2 optionnel est un port TWDNAC232D.

Pour configurer l'automate, connectez le câble TSXPCX1031 (non illustré) au port 1 de l'automate Twido. Connectez ensuite le câble au port COM 1 du PC. Vérifiez que le commutateur est en position 2. Enfin, connectez le port COM 2 du PC au port 2 EIA RS-232 de l'automate Twido. Les connexions des broches et le câblage sont présentés dans l'étape suivante.

Etape 2 : Connexion du câble de communication ASCII (EIA RS-232)



La configuration minimale requise pour le câblage du câble de communication ASCII correspond à une connexion à 3 fils de base. Croisez les signaux de transmission et de réception.

Note : A l'extrémité PC du câble, des connexions supplémentaires (telles que DTR et DSR) peuvent être nécessaires afin de satisfaire le protocole de transmission. Aucune connexion supplémentaire n'est requise pour satisfaire l'automate Twido.

Etape 3 : Configuration du port

Matériel -> Ajouter une option TWDNOZ232D

Matériel => Comm. automate Réglage	
Port :	2
Type :	ASCII
Débit :	19 200
Données :	8 bits
Parité :	Aucune
Arrêt :	1 bit
Fin de trame : 65	
Délai de réponse : 100 x 100 ms	

Emulateur de terminal sur un PC	
Port :	COM2
Débit :	19 200
Données :	8 bits
Parité :	Aucune
Arrêt :	1 bit
Régulation du flux :	Aucune

Utilisez une simple application d'émulateur de terminal sur le PC pour configurer une configuration de port de base et pour garantir l'absence de régulation de flux. Utilisez TwidoSoft pour configurer le port de l'automate. En premier lieu, configurez l'option matérielle. Dans cet exemple, le port TWDNOZ232D est ajouté à la base automate modulaire.

En second lieu, initialisez le paramétrage de la communication de l'automate à l'aide des mêmes paramètres que ceux de l'émulateur de terminal sur le PC. Dans cet exemple, la lettre majuscule "A" est choisie comme caractère de "fin de trame", afin de terminer le tampon de réception d'entrée. Un délai de dix secondes est choisi pour le paramètre Délai de réponse. Un seul de ces deux paramètres sera appelé, selon celui qui se produira en premier.

Etape 4 : Ecriture d'une application

```
LD 1
[%MW10 := 16#0104]
[%MW11 := 16#0000]
[%MW12 := 16#4F4B]
[%MW13 := 16#0A0D]
LD 1
AND %MSG2.D
[EXCH2 %MW10:8]
LD %MSG2.E
ST %Q0.0
END
```

Utilisez TwidoSoft pour créer un programme d'application en trois temps. Tout d'abord, initialisez la régulation et le tampon de transfert à utiliser pour l'instruction EXCH. Dans cet exemple, une commande est configurée pour à la fois envoyer et recevoir des données. La quantité de données à envoyer est réglée sur quatre octets et initialisée sur les caractères : "O", "K", CR et LF.

Vérifiez ensuite le bit terminé associé à %MSG2 et émettez l'instruction EXCH2 uniquement si le port est prêt. Une valeur de huit caractères est spécifiée pour l'instruction EXCH2. Il existe deux mots de commande (%MW10 et %MW11), deux mots à utiliser pour les informations de transmission (%MW12 et %MW13) et quatre mots pour recevoir des données (%MW14 à %MW17).

Enfin, l'état d'erreur de %MSG2 est détecté et stocké sur le premier bit de sortie de l'E/S de la base automate locale. Il est également possible d'ajouter une recherche d'erreurs supplémentaire à l'aide de %SW64 pour rendre celle-ci plus robuste.

Etape 5 : Initialisation de l'éditeur de tables d'animation

Format courant conservé du repère			
1 %MW10	0104	0000	Hexadécimal
2 %MW10	0104	0000	Hexadécimal
3 %MW10	0104	0000	Hexadécimal
4 %MW13	0A0D	0000	Hexadécimal
5 %MW14	TW	0000	ASCII
6 %MW15	ID	0000	ASCII
7 %MW16	O	0000	ASCII
8 %MW17	A	0000	ASCII

L'étape finale consiste à télécharger cette application d'automate et à l'exécuter. Initialisez l'éditeur de tables d'animation pour animer et afficher les mots %MW10 à %MW17. Sur l'émulateur de terminal, les caractères "O"-"K"-CR-LF s'affichent. De nombreux caractères peuvent s'afficher en fonction du nombre de fois que le délai du bloc EXCH s'est écoulé et qu'un nouveau bloc a été émis. Sur l'émulateur de terminal, tapez "T"-"W"-"I"-"D"-"O"-" "-"A". Ces informations sont échangées avec l'automate Twido et s'affichent dans l'éditeur de tables d'animation.

Communications Modbus

Introduction

Le protocole Modbus est un protocole maître-esclave qui permet à un seul et unique maître de demander des réponses à des esclaves ou d'agir en fonction de la requête. Le maître peut s'adresser aux esclaves individuellement ou envoyer un message de diffusion générale à tous les esclaves. Les esclaves renvoient un message (réponse) aux requêtes qui leur sont adressées individuellement. Les réponses aux requêtes à diffusion générale ne sont pas renvoyées par le maître.

Configuration matérielle

Il est possible d'établir une liaison Modbus sur le port EIA RS-232 ou EIA RS-485 et de l'exécuter simultanément sur deux ports de communication au maximum. Le tableau suivant répertorie les périphériques utilisables.

Périphérique	Port	Caractéristiques
TWDCAA10/16/24DRF, TWDLMDA20/40DUK, TWDLMDA20/40DTK, TWDLMDA20DRT	1	Base automate prenant en charge un port EIA RS-485 à 3 fils à l'aide d'un connecteur mini DIN.
TWDNOZ232D	2	Module de communication prenant en charge un port EIA RS-232 à 3 fils à l'aide d'un connecteur mini DIN. **Note :** Ce module n'est disponible que pour les automates modulaires. Lorsque le module est connecté, l'automate ne peut pas disposer d'un module d'expansion Afficheur.
TWDNOZ485D	2	Module de communication prenant en charge un port EIA RS-485 à 3 fils à l'aide d'un connecteur mini DIN. **Note :** Ce module n'est disponible que pour les automates modulaires. Lorsque le module est connecté, l'automate ne peut pas disposer d'un module d'expansion Afficheur.
TWDNOZ485T	2	Module de communication prenant en charge un port EIA RS-485 à 3 fils à l'aide d'un connecteur de borne. **Note :** Ce module n'est disponible que pour les automates modulaires. Lorsque le module est connecté, l'automate ne peut pas disposer d'un module d'expansion Afficheur.
TWDNAC232D	2	Adaptateur de communication prenant en charge un port EIA RS-232 à 3 fils à l'aide d'un connecteur mini DIN. **Note :** Cet adaptateur n'est disponible que pour les automates 16 et 24 E/S compacts et pour le module d'expansion Afficheur.
TWDNAC485D	2	Adaptateur de communication prenant en charge un port EIA RS-485 à 3 fils à l'aide d'un connecteur mini DIN. **Note :** Cet adaptateur n'est disponible que pour les automates 16 et 24 E/S compacts et pour le module d'expansion Afficheur.
TWDNAC485T	2	Adaptateur de communication prenant en charge un port EIA RS-485 à 3 fils à l'aide d'un connecteur de borne. **Note :** Cet adaptateur n'est disponible que pour les automates 16 et 24 E/S compacts et pour le module d'expansion Afficheur.

Périphérique	Port	Caractéristiques
TWDXCPODM	2	Module d'expansion Afficheur prenant en charge un port EIA RS-232 à 3 fils à l'aide d'un connecteur mini DIN, un port EIA RS-485 à 3 fils à l'aide d'un connecteur mini DIN ou un port EIA RS-485 à 3 fils à l'aide d'un connecteur de borne. **Note :** Ce module n'est disponible que pour les automates modulaires. Lorsque le module est connecté, l'automate ne peut pas disposer d'un module d'expansion Communication.

Note : La configuration du port 2 (disponibilité et type) est uniquement contrôlée lors de la mise sous tension ou de la réinitialisation par le microprogramme de l'automate.

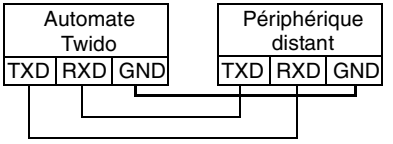
Câblage nominal

Les connexions de câble nominal sont représentées ci-dessous pour les types EIA RS-232 et EIA RS-485.

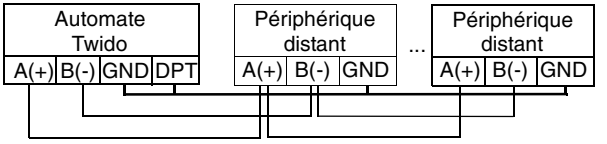
Note : Si vous utilisez le port 1 de l'automate Twido, vous devez connecter le signal DPT à GND (terre). Ce signal permet d'indiquer à l'automate Twido que les communications via le port 1 relèvent du protocole Modbus et non du protocole utilisé pour communiquer avec le logiciel TwidoSoft. Ce périphérique distant spécifique peut requérir l'utilisation de signaux supplémentaires (DTR, DSR, etc.).

Les connexions de câbles effectués à chaque périphérique sont représentées ci-dessous.

Câble EIA RS-232



Câble EIA RS-485



Note : La connexion DPT à GND (terre) n'est nécessaire qu'en cas de connexion à une base automate sur le port 1.

Configuration logicielle

Pour configurer l'automate afin d'utiliser une connexion en série pour envoyer et recevoir des caractères à l'aide du protocole Modbus, vous devez procéder comme suit :

Etape	Description
1	Configurez le port série pour le protocole Modbus à l'aide de TwidoSoft.
2	Créez dans votre application un tampon de transmission/réception pour que le protocole Modbus utilise l'instruction EXCHx.

Configuration du port

Un automate Twido peut utiliser son port 1 principal ou un port 2 configuré en option pour utiliser le protocole Modbus. Pour configurer un port série pour le protocole Modbus :

Etape	Action
1	Définissez tous les modules ou cartouches optionnels supplémentaires physiquement configurés sur la base.
2	Cliquez avec le bouton droit sur le port, puis cliquez sur Editer le paramétrage communication de l'automate... et modifiez le type du port série par "Modbus".
3	Définissez les paramètres de communication associés.

Modbus maître

Le mode Modbus maître permet à l'automate d'émettre la transmission d'une requête Modbus et d'en attendre une réponse d'un esclave Modbus. Le mode Modbus maître n'est pris en charge que par l'intermédiaire de l'instruction EXCHx. Les modes Modbus ASCII et RTU sont tous les deux pris en charge en mode Modbus maître.
La taille maximale des trames transmises et/ou reçues s'élève à 128 octets. En outre, la table de mots associée à l'instruction EXCHx se compose des tables de transmission et de réception.

	Octet le plus significatif	**Octet le moins significatif**
Mots de commande	Commande	Longueur (Transmetteur/ Récepteur)
	Décalage récepteur	Décalage transmetteur
Table de transmission	Octet 1 transmis	Octet 2 transmis
	...	...
	...	Octet n transmis
	Octet n+1 transmis	
Table de réception	Octet 1 reçu	Octet 2 reçu
	...	...
	...	Octet p reçu
	Octet p+1 reçu	

Paramètres de contrôle

L'octet **Longueur** comprend la longueur à transmettre, qui est écrasée par le nombre de caractères reçus à la fin de la réception, si la réception est requise. Ce paramètre correspond à la longueur en octets de la table de transmission. Si le paramètre de décalage du transmetteur est égal à zéro, ce paramètre sera égal à la longueur même de la trame moins les deux octets CRC. Si le paramètre de décalage du transmetteur n'est pas égal à zéro, un octet du tampon (indiqué par la valeur de décalage) ne sera pas transmis et ce paramètre sera égal à la longueur même de la trame plus 1.

L'octet **Commande** doit toujours être égal à 1 (transmetteur et récepteur) en cas de requête Modbus RTU (sauf pour une diffusion générale).

L'octet **Décalage transmetteur** contient le décalage (1 pour le premier octet, 2 pour le deuxième octet, etc.) dans la table de transmission à ignorer lors de la transmission du paquet. Il est utilisé pour prendre en charge les émissions associées aux valeurs octet/mot dans le cadre du protocole Modbus. Par exemple, si cet octet est égal à 3, le troisième octet est ignoré, ce qui fait du quatrième octet de la table le troisième octet à transmettre.

L'octet **Décalage récepteur** contient le décalage (1 pour le premier octet, 2 pour le deuxième octet, etc.) dans la table de réception à ajouter lors de la transmission du paquet. Il est utilisé pour prendre en charge les émissions associées aux valeurs octet/mot dans le cadre du protocole Modbus. Par exemple, si cet octet est égal à 3, le troisième octet de la table est renseigné par un ZERO et le troisième octet réellement reçu est entré dans le quatrième emplacement de la table.

Tables de transmission/ réception

Dans l'un ou l'autre des modes (Modbus ASCII ou Modbus RTU), la table de transmission est renseignée à l'aide de la requête avant l'exécution de l'instruction EXCHx. Au moment de l'exécution, l'automate détermine quelle est la couche liaison de données et effectue toutes les conversions nécessaires pour traiter la transmission et la réponse. Les caractères de début, de fin et de contrôle ne sont pas stockés dans les tables de transmission/réception.
Une fois que tous les octets ont été transmis, l'automate passe en mode de réception et est prêt à recevoir des octets. La réception s'accomplit de l'une des manières suivantes : le caractère de fin de trame est reçu en mode ASCII ; un délai a été détecté sur un caractère ou une trame ; la table de réception est pleine.
Les entrées **Octet transmis X** contiennent les données (codage RTU) de protocole Modbus à transmettre. Si le port de communication est configuré en Modbus ASCII, les caractères de trame corrects sont ajoutés à la transmission. Le premier octet comprend le repère du périphérique (spécifique ou général), le deuxième octet comprend le code de fonction et le reste comprend les informations associées à ce code de fonction.

Note : Il s'agit d'une application typique, mais toutes les possibilités ne sont pas définies. Aucune validation des données en cours de transmission n'est effectuée.

Les entrées **Octet reçu X** contiennent les données (codage RTU) de protocole Modbus à recevoir. Si le port de communication est configuré en Modbus ASCII, les caractères de trame corrects sont supprimés de la réponse. Le premier octet comprend le repère du périphérique, le deuxième octet comprend le code de fonction (ou code de réponse) et le reste comprend les informations associées à ce code de fonction.

Note : Il s'agit d'une application typique, mais toutes les possibilités ne sont pas définies. Aucune validation des données en cours de réception n'est effectuée, à l'exception d'une vérification de checksum.

Modbus esclave

Le mode Modbus esclave permet à l'automate de répondre à des requêtes Modbus à partir d'un maître Modbus. L'automate prend en charge les fonctions de régulation et de données Modbus standard, ainsi que les extensions UMAS pour la configuration et l'accès aux objets.
Lorsque le câble TSXPCX1031 est raccordé à l'automate, la communication en mode Modbus esclave démarre au niveau du port, ce qui désactive temporairement le mode de communication qui était en cours d'exécution avant la connexion de ce câble.
Le protocole Modbus prend en charge deux formats de couche liaison de données : ASCII et RTU. Chaque format est défini par l'implémentation de la couche physique ; le format ASCII utilise sept bits de données tandis que le format RTU en utilise huit.
En mode Modbus ASCII, chaque octet d'un message est envoyé sous la forme de deux caractères ASCII. La trame Modbus ASCII commence par un caractère de début (':') et se termine par deux caractères de fin (CR et LF). Le caractère de fin de trame par défaut est 0x0A (LF). L'utilisateur peut modifier la valeur de cet octet au cours de la configuration. La valeur de contrôle de la trame Modbus ASCII correspond à un simple complément de deux de la trame, excluant les caractères de début et de fin.
Le mode Modbus RTU ne reformate pas le message avant de le transmettre ; cependant, il utilise un mode de calcul de checksum différent, spécifié sous forme de CRC.
Les limitations de la couche liaison de données Modbus sont les suivantes :

- Repère 1-247
- Bits : 128 bits sur demande à l'aide de requêtes ouvertes Modbus
- Mots : 64 mots de 16 bits sur demande à l'aide de requêtes ouvertes Modbus

Echange de messages

Il est possible de configurer l'automate Twido pour envoyer et/ou recevoir des messages en mode caractère. Le langage propose deux services pour celui-ci :

- **Instruction EXCHx :** pour transmettre/recevoir des messages
- **Bloc fonction %MSGx :** pour contrôler les échanges de messages.

L'automate Twido utilise le protocole configuré pour ce port lors du traitement d'une instruction EXCHx.

Note : Il est possible de configurer chaque port de communication pour différents protocoles ou pour le même protocole. Pour accéder à l'instruction EXCHx ou au bloc fonction %MSGx de chaque port de communication, il suffit d'y ajouter le numéro du port (1 ou 2).

Instruction EXCHx

L'instruction EXCHx permet à l'automate Twido d'envoyer et/ou de recevoir des informations vers/depuis des périphériques Modbus. L'utilisateur définit une table de mots (%MWi:L ou %KWi:L) contenant des informations de contrôle ainsi que les données à envoyer et/ou à recevoir (jusqu'à 64 mots de données dans la transmission et/ou réception). La description du format de la table de mots a été donnée précédemment.

Un échange de messages s'effectue à l'aide de l'instruction EXCHx.

Syntaxe : [EXCHx %MWi:L] ou [EXCHx %KWi:L]

où : x = numéro du port (1 ou 2)

L = nombre de mots du tableau des mots

L'automate Twido doit terminer l'échange de la première instruction EXCHx avant de pouvoir en lancer une deuxième. Il est nécessaire d'utiliser le bloc fonction %MSGx lors de l'envoi de plusieurs messages.

Le traitement de l'instruction par liste EXCHx se produit immédiatement, en sachant que toutes les transmissions sont démarrées sous contrôle d'interruptions (la réception des données est également sous contrôle d'interruptions), ce qui est considéré comme un traitement en arrière-plan.

Bloc fonction %MSGx

L'utilisation du bloc fonction %MSGx est facultative ; il permet de gérer des échanges de données. Le bloc fonction %MSGx remplit trois fonctions :

- **Vérification des erreurs de communications**
  La recherche d'erreurs permet de vérifier que la longueur du bloc (table de mots) programmée à l'aide de l'instruction EXCHx est suffisamment grande pour contenir la longueur du message à envoyer. Celle-ci est comparée à la longueur programmée dans l'octet le moins significatif du premier mot de la table de mots.
- **Coordination de plusieurs messages**
  Pour garantir la coordination lors de l'envoi de plusieurs messages, le bloc fonction %MSGx fournit les informations requises pour déterminer le moment où la transmission du message précédent est terminée.
- **Transmission de messages prioritaires**
  Le bloc fonction %MSGx vous permet de suspendre la transmission d'un message afin d'envoyer un message plus urgent.

Le bloc fonction %MSGx dispose d'une entrée et de deux sorties qui lui sont associées :

Entrée/Sortie	Définition	Description
R	Entrée RAZ	Réglée sur 1 : réinitialise la communication ou le bloc (%MSGx.E = 0 et %MSGx.D = 1).
%MSGx.D	Communication terminée	0: requête en cours. 1: communication terminée en cas de fin de transmission, de réception du caractère de fin, d'erreur ou de réinitialisation du bloc.
%MSGx.E	erreur	0: longueur du message OK et liaison OK. 1: en cas de mauvaise commande, de table configurée de manière incorrecte, de mauvais caractère reçu (débit, parité, etc.) ou de saturation de la table de réception.

Limitations

Il est important de garder à l'esprit les limitations suivantes :

- La disponibilité et le type du port 2 sont uniquement contrôlés lors de la mise sous tension ou de la réinitialisation.
- Tout message en cours de traitement sur le port 1 est abandonné lorsque TwidoSoft est connecté.
- Il est impossible de traiter EXCHx ou %MSG sur un port configuré en tant que liaison distante.
- EXCHx abandonne le traitement Modbus esclave actif (à l'exception du traitement TwidoSoft).
- Le traitement des instructions EXCHx ne fait pas l'objet d'une nouvelle tentative en cas d'erreur.
- Il est possible d'utiliser R %MSGx pour annuler le traitement de la réception d'une instruction EXCHx.
- Il est possible de configurer des instructions EXCHx avec un délai d'annulation de réception.
- Les messages multiples sont contrôlés via %MSGx.D.

Conditions d'erreur et de mode de fonctionnement

Si une erreur se produit lors de l'utilisation de l'instruction EXCHx, les bits %MSGx.D et %MSGx.E sont réglés sur 1, le mot système %SW63 contient le code d'erreur du port 1 et %SW64 le code d'erreur du port 2.

Mots système	Utilisation
%SW63	Code d'erreur EXCH1 : 0 - opération réussie 1 - tampon de transmission trop important (> 128) 2 - tampon de transmission trop petit 3 - table de mots trop petite 4 - débordement de la table de réception 5 - délai écoulé 6 - erreur de transmission (erreur reçue en réponse) 7 - mauvaise commande dans la table 8 - port sélectionné non configuré/disponible 9 - erreur de réception 10 - impossible d'utiliser %KW en cas de réception 11 - décalage de transmission plus important que la table de transmission 12 - décalage de réception plus important que la table de réception 13 - interruption du traitement EXCH par l'automate
%SW64	Code d'erreur EXCH2 : Voir %SW63.

Redémarrage de l'automate maître

Lorsqu'un automate maître/esclave redémarre, l'un des événements suivants se produit :

- Un démarrage à froid (%S0 = 1) force la réinitialisation des communications.
- Un démarrage à chaud (%S1 = 1) force la réinitialisation des communications.
- En mode Stop, l'automate arrête toutes les communications Modbus.

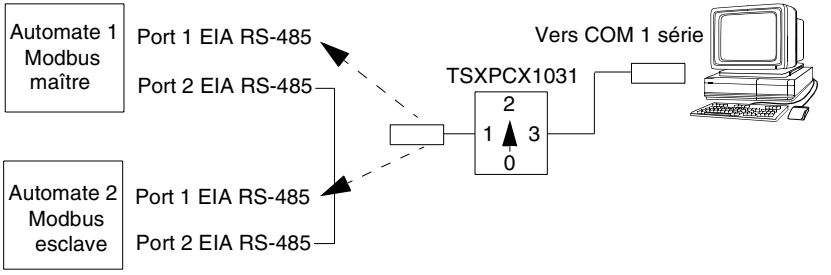
Exemple 1 de liaison Modbus

Pour configurer une liaison Modbus, vous devez procéder comme suit :

1. Configurez le matériel.
2. Connectez le câble de communication Modbus.
3. Configurez le port.
4. Ecrivez une application.
5. Initialisez l'éditeur de tables d'animation.

Les illustrations suivantes représentent l'utilisation du code de fonction Modbus 3 pour lire des mots de sortie d'un esclave. Cet exemple utilise deux automates Twido.

Etape 1 : Configuration du matériel



La configuration matérielle comprend deux automates Twido. L'un d'entre eux est configuré en tant que Modbus maître et l'autre en tant que Modbus esclave.

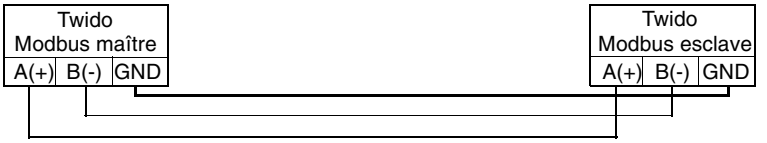
Note : Dans cet exemple, chaque automate est configuré afin d'utiliser EIA RS-485 sur le port 1 ainsi que EIA RS-485 sur le port 2 optionnel. Sur un automate modulaire, le port 2 optionnel peut être de type TWDNOZ485D ou TWDNOZ485T. Sur un automate compact, le port 2 optionnel peut être un port TWDNAC485D ou TWDNAC485T.

Pour configurer chaque automate, connectez le câble TSXPCX1031 au port 1 du premier automate.

Note : Le câble TSXPCX1031 peut uniquement être connecté à un automate à la fois, sur le port 1 EIA RS-485 uniquement.

Connectez ensuite le câble au port COM 1 du PC. Assurez-vous que le commutateur est en position 2. Téléchargez et contrôlez chaque application. Répétez cette procédure pour le deuxième automate.

Etape 2 : Connexion du câble de communication Modbus



Le câblage utilisé dans cet exemple correspond à une simple connexion point à point. Les trois signaux A(+), B(-) et GND sont câblés conformément à l'illustration. En cas d'utilisation du port 1 de l'automate Twido, le signal DPT doit être relié à la terre. Cette condition du DPT détermine si TwidoSoft est connecté. Lorsqu'il est relié à la terre, l'automate utilise la configuration de port définie dans l'application pour déterminer le type de communication.

Etape 3 : Configuration du port

Matériel -> Ajouter une option TWDNOZ485-
Matériel => Comm. automate Réglage Port : 2 Type : Modbus Repère : 1 Débit : 19 200 Données : 8 bits Parité : Aucune Arrêt : 1 bit Fin de trame : 65 Délai de réponse : 10 x 100 ms Dépassement trame : 10 ms

Matériel -> Ajouter une option TWDNOZ485-
Matériel => Comm. automate Réglage Port : 2 Type : Modbus Repère : 2 Débit : 19 200 Données : 8 bits Parité : Aucune Arrêt : 1 bit Fin de trame : 65 Délai de réponse : 100 x 100 ms Dépassement trame : 10 ms

Dans les applications maître et esclave, les ports EIA RS-485 optionnels sont configurés. Assurez-vous de modifier les communications de l'automate pour émettre les repères Modbus ou le port 2 vers deux repères différents. Dans cet exemple, le maître est réglé sur un repère de 1 et l'esclave sur 2. Le nombre de bits est réglé sur 8, ce qui indique que le mode Modbus RTU sera utilisé. S'il avait été réglé sur 7, le mode Modbus ASCII aurait été utilisé. La seule autre valeur par défaut modifiée concerne l'augmentation du délai de réponse à 1 seconde.

Note : Etant donné que le mode Modbus RTU a été sélectionné, le paramètre "Fin de trame" a été ignoré.

Etape 4 : Ecriture d'une application

```
LD 1
[%MW0 := 16#0106]
[%MW1 := 16#0300]
[%MW2 := 16#0203]
[%MW3 := 16#0000]
[%MW4 := 16#0004]
LD 1
AND %MSG2.D
[EXCH2 %MW0:11]
LD %MSG2.E
ST %Q0.0
END
```

```
LD 1
[%MW0 := 16#6566]
[%MW1 := 16#6768]
[%MW2 := 16#6970]
[%MW3 := 16#7172]
END
```

A l'aide de TwidoSoft, un programme d'application est écrit pour le maître et l'esclave. Pour l'esclave, il suffit d'émettre certains mots mémoire vers un ensemble de valeurs connues. Dans le maître, le bloc d'échange est initialisé afin de lire quatre mots de l'esclave au niveau du repère Modbus 2 qui démarre à l'emplacement %MW0.

Note : Remarquez l'utilisation du décalage récepteur défini dans %MW1 du maître Modbus. Le décalage de trois ajoute un octet (valeur = 0) à la troisième position de la zone de réception de la table. Il permet d'aligner les mots dans le maître de façon à ce qu'ils entrent correctement dans les limites de mot. Sans ce décalage, chaque mot de données serait fractionné en deux mots dans le bloc d'échange. Ce décalage est utilisé pour des raisons de commodité.

Avant d'émettre l'instruction EXHC2, l'application vérifie le bit terminé associé à %MSG2. Enfin, l'état d'erreur de %MSG2 est détecté et stocké sur le premier bit de sortie de l'E/S de la base automate locale. Il est également possible d'ajouter une recherche d'erreurs supplémentaire à l'aide de %SW64 pour rendre celle-ci plus robuste.

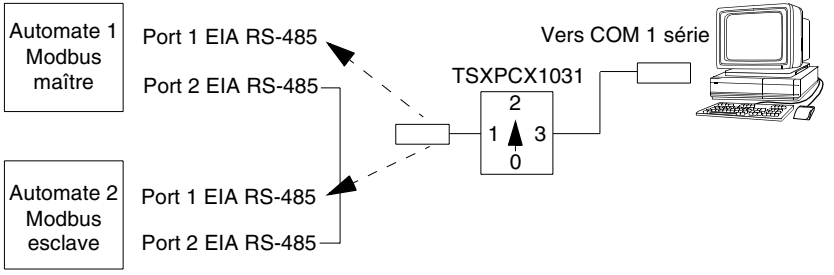
Etape 5 : Initialisation de l'éditeur de tables d'animation

Format courant conservé du repère			
1 %MW5	0203	0000	Hexadécimal
2 %MW6	0008	0000	Hexadécimal
3 %MW7	6566	0000	Hexadécimal
4 %MW8	6868	0000	Hexadécimal
5 %MW9	6970	0000	Hexadécimal
6 %MW10	7172	0000	Hexadécimal

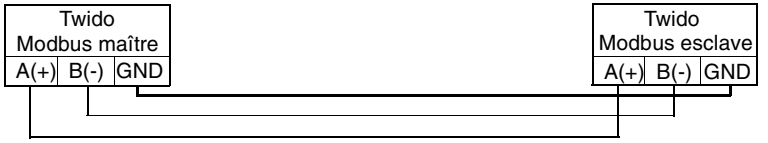
Après le téléchargement et la configuration de tous les automates en vue de leur exécution, ouvrez une table d'animation sur le maître. Examinez la section réponse de la table pour vérifier que le code de réponse correspond à 3 et que le nombre d'octets lus est correct. Notez également, dans cet exemple, que les mots lus de l'esclave (commençant par %MW7) sont correctement alignés avec les limites de mot dans le maître.

Exemple 2 de liaison Modbus

L'illustration suivante représente l'utilisation du code de fonction Modbus 16 pour écrire des mots de sortie sur un esclave. Cet exemple utilise deux automates Twido.
Etape 1 : Configuration du matériel



La configuration matérielle est identique à celle de l'exemple précédent.
Etape 2 : Connexion du câble de communication Modbus



Le câblage de communication Modbus est identique à celui de l'exemple précédent.
Etape 3 : Configuration du port

Matériel -> Ajouter une option TWDNOZ485-

Matériel => Comm. automate Réglage	
Port :	2
Type :	Modbus
Repère :	1
Débit :	19200
Données :	8 bits
Parité :	Aucune
Arrêt :	1 bit
Fin de trame :	65
Délai de réponse : 10 x 100 ms	
Dépassement trame : 10 ms	

Matériel -> Ajouter une option TWDNOZ485-

Matériel => Comm. automate Réglage	
Port :	2
Type :	Modbus
Repère :	2
Débit :	19200
Données :	8 bits
Parité :	Aucune
Arrêt :	1 bit
Fin de trame :	65
Délai de réponse : 100 x 100 ms	
Dépassement trame : 10 ms	

Les configurations du port sont identiques à celles de l'exemple précédent.

Etape 4 : Ecriture d'une application

```
LD 1
[%MW0 := 16#010C]
[%MW1 := 16#0007]
[%MW2 := 16#0210]
[%MW3 := 16#0010]
[%MW4 := 16#0002]
[%MW5 := 16#0004]
[%MW6 := 16#6566]
[%MW7 := 16#6768]
LD 1
AND %MSG2.D
[EXCH2 %MW0:11]
LD %MSG2.E
ST %Q0.0
END
```

```
LD 1
[%MW18 := 16#FFFF]
END
```

A l'aide de TwidoSoft, un programme d'application est créé pour le maître et l'esclave. Pour l'esclave, émettez un seul mot mémoire %MW18. Cette action permet d'allouer de l'espace sur l'esclave pour les repères mémoire de %MW0 à %MW18. Sans allocation d'espace, le bloc d'échange essaierait d'écrire à des emplacements inexistants sur l'esclave.
Dans le maître, le bloc d'échange est émis afin d'écrire douze mots (0C hexadécimal) vers l'esclave au niveau du repère Modbus 2 qui démarre à l'emplacement %MW16 (10 hexadécimal).

Note : Remarquez l'utilisation du décalage transmetteur défini dans %MW1 de l'application du maître Modbus. Le décalage de sept permet de supprimer l'octet le plus significatif dans le sixième mot (valeur 00 hexadécimale dans %MW5). Cette action permet d'aligner les valeurs de données dans la table de transmission du bloc d'échange de façon à ce qu'elles entrent correctement dans les limites de mot.

Avant d'émettre l'instruction EXHC2, l'application vérifie le bit terminé associé à %MSG2. Enfin, l'état d'erreur de %MSG2 est détecté et stocké sur le premier bit de sortie de l'E/S de la base automate locale. Il est également possible d'ajouter une recherche d'erreurs supplémentaire à l'aide de %SW64 pour rendre celle-ci plus robuste.

Etape 5 : Initialisation de l'éditeur de tables d'animation

Format courant conservé du repère			
1 %MW0	010C	0000	Hexadécimal
2 %MW1	0007	0000	Hexadécimal
3 %MW2	0210	0000	Hexadécimal
4 %MW3	0010	0000	Hexadécimal
5 %MW4	0002	0000	Hexadécimal
6 %MW5	0004	0000	Hexadécimal
7 %MW6	6566	0000	Hexadécimal
8 %MW7	6768	0000	Hexadécimal
9 %MW8	0210	0000	Hexadécimal
10 %MW9	0010	0000	Hexadécimal
11 %MW10	0004	0000	Hexadécimal

Format courant conservé du repère			
1 %MW16	6566	0000	Hexadécimal
2 %MW17	6768	0000	Hexadécimal

Après le téléchargement et la définition de tous les automates de sorte à ce qu'ils s'exécutent, ouvrez une table d'animation. Les deux valeurs de %MW16 et %MW17 sont écrites sur l'esclave. Dans le maître, il est possible d'utiliser la table d'animation afin d'examiner la partie table de réception des données d'échange. Ces données affichent le repère esclave, le code de réponse, le premier mot écrit et le nombre de mots écrits à partir de %MW8 dans l'exemple ci-dessus.

Requêtes Modbus standard

Introduction

Ces requêtes permettent d'échanger des données entre les périphériques afin d'accéder à des informations de bit ou de mot. Le format de table utilisé est le même pour le mode RTU et pour le mode ASCII.

Format	Référence
Bit	%Mi, registres 0x ou 1x
Mot	%MWi, registres 3x ou 4x

Maître Modbus : Lecture de N bits d'entrée et de sortie

La table suivante représente les requêtes 01 et 02.

	Index de la table	Octet le plus significatif	Octet le moins significatif
Contrôle	0	01 (Transmetteur/ Récepteur)	06 (Longueur Transmetteur)
	1	00 (Décalage Récepteur)	00 (Décalage Transmetteur)
Table de transmission	2	Esclave@(1..247)	01 (Code de requête)
	3	Numéro du premier bit à lire	
	4	N = Nombre de bits à lire	
Table de réception (après réponse)	5	Esclave@(1..247)	01 (Code de réponse)
	6	Nombre d'octets des données transmis (un octet par bit)	
	7	Premier octet lu (valeur = 00 ou 01)	Deuxième octet lu (si N>1)
	8	Troisième octet lu	
	...		
	(N/2)+6	Nième octet lu (si N>1)	

Maître Modbus : Lecture de N mots d'entrée et de sortie

La table suivante représente les requêtes 03 et 04.

	Index de la table	**Octet le plus significatif**	**Octet le moins significatif**
Contrôle	0	01 (Transmetteur/ Récepteur)	06 (Longueur Transmetteur)
	1	03 (Décalage Récepteur)	00 (Décalage Transmetteur)
Table de transmission	2	Esclave@(1..247)	03 (Code de requête)
	3	Numéro du premier mot à lire	
	4	N = Nombre de mots à lire	
Table de réception (après réponse)	5	Esclave@(1..247)	03 (Code de réponse)
	6	00 (octet ajouté à la suite d'une action de Décalage Récepteur)	2*N (nombre d'octets lus)
	7	Premier mot lu	
	8	Deuxième mot lu (si N>1)	
	...		
	N+6	Nième mot lu (si N>2)	

Note : L'opération Décalage Récepteur = 3 ajoute un octet (valeur = 0) à la troisième position de la table de réception. Elle assure également un bon positionnement dans la table, du nombre d'octets lus et des valeurs des mots lus.

Maître Modbus : Ecriture d'un bit de sortie

La table suivante représente la requête 05.

	Index de la table	**Octet le plus significatif**	**Octet le moins significatif**
Contrôle	0	01 (Transmetteur/ Récepteur)	06 (Longueur Transmetteur)
	1	00 (Décalage Récepteur)	00 (Décalage Transmetteur)
Table de transmission	2	Esclave@(1..247)	05 (Code de requête)
	3	Numéro du bit à écrire	
	4	Valeur du bit à écrire	
Table de réception (après réponse)	5	Esclave@(1..247)	05 (Code de réponse)
	6	Numéro du bit écrit	
	7	Valeur écrite	

Note :
- Il n'est pas nécessaire d'utiliser le décalage pour cette requête.
- La trame de la réponse est identique à celle de cette requête (dans un cas normal).
- Pour affecter la valeur 1 à un bit, le mot associé dans la table de transmission doit contenir la valeur FF00H, et 0 pour affecter la valeur 0 à un bit.

Maître Modbus : Ecriture d'un mot de sortie

La table suivante représente la requête 06.

	Index de la table	Octet le plus significatif	Octet le moins significatif
Contrôle	0	01 (Transmetteur/ Récepteur)	06 (Longueur Transmetteur)
	1	00 (Décalage Récepteur)	00 (Décalage Transmetteur)
Table de transmission	2	Esclave@(1..247)	06 (Code de requête)
	3	Numéro du mot à écrire	
	4	Valeur du mot à écrire	
Table de réception (après réponse)	5	Esclave@(1..247)	06 (Code de réponse)
	6	Numéro du mot écrit	
	7	Valeur écrite	

Note :
- Il n'est pas nécessaire d'utiliser le décalage pour cette requête.
- La trame de la réponse est identique à celle de cette requête (dans un cas normal).

Maître Modbus : Ecriture de N bits de sortie

La table suivante représente la requête 15.

	Index de la table	**Octet le plus significatif**	**Octet le moins significatif**
Contrôle	0	01 (Transmetteur/ Récepteur)	8 + nombre d'octets (Transmetteur)
	1	00 (Décalage Récepteur)	07 (Décalage Transmetteur)
Table de transmission	2	Esclave@(1..247)	15 (Code de requête)
	3	Numéro du premier bit à écrire	
	4	N_1 = Nombre de bits à écrire	
	5	00 (octet non envoyé, effet de décalage)	N_2 = Nombre d'octets des données à écrire
	6	Valeur du premier octet	Valeur du premier octet
	7	Valeur du troisième octet	
	...		
	6+(N_2/2)	Valeur du N_2ième octet	
Table de réception (après réponse)		Esclave@(1..247)	15 (Code de réponse)
		Numéro du premier bit écrit	
		Nombre de bits écrits (= N_1)	

Note :

- L'opération Décalage Transmetteur = 7 supprime le 7ième octet de la trame envoyée. Elle permet également d'assurer une bonne correspondance entre les valeurs des mots de la table de transmission.

Maître Modbus : Ecriture de N mots de sortie

La table suivante représente la requête 16.

	Index de la table	Octet le plus significatif	Octet le moins significatif
Contrôle	0	01 (Transmetteur/ Récepteur)	8 + (2*N) (Longueur Transmetteur)
	1	00 (Décalage Récepteur)	07 (Décalage Transmetteur)
Table de transmission	2	Esclave@(1..247)	16 (Code de requête)
	3	Numéro du premier mot à écrire	
	4	N = Nombre de mots à écrire	
	5	00 (octet non envoyé, effet de décalage)	2*N = Nb d'octets à écrire
	6	Première valeur du mot à écrire	
	7	Deuxième valeur à écrire	
	...		
	N+5	N valeurs à écrire	
Table de réception (après réponse)	N+6	Esclave@(1..247)	16 (Code de réponse)
	N+7	Numéro du premier mot écrit	
	N+8	Nombre de bits écrits (= N)	

Note : L'opération Décalage Transmetteur = 7 supprime le 5ème octet MMSB de la trame envoyée. Elle permet également d'assurer une bonne correspondance entre les valeurs des mots de la table de transmission.

Fonctions analogiques intégrées

6

En bref...

Présentation

Cette rubrique décrit la gestion de la voie analogique et des points de réglage analogique intégrés.

Contenu de ce chapitre

Ce chapitre contient les sujets suivants :

Points de réglage analogique

Introduction

Les automates Twido possèdent :
- un point de réglage analogique sur les automates TWDLCAA10DRF et TWDLCAA16DRF
- deux points de réglage analogique sur l'automate TWDLCAA24DRF.

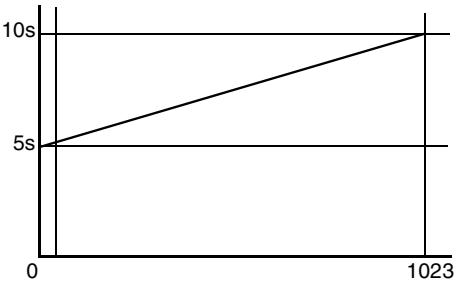
Programmation

Les valeurs numériques, allant de 0 à 1023 pour le point de réglage analogique 1 et de 0 à 511 pour le point de réglage analogique 2, et correspondant aux valeurs analogiques données par ces points de réglage analogique sont contenues dans les deux mots système suivants :
- %IW0.0.0 pour le point de réglage analogique 1 (situé le plus à gauche)
- %IW0.0.1 pour le point de réglage analogique 2 (situé le plus à droite)

Ces mots peuvent être utilisés dans des opérations arithmétiques et pour n'importe quel type de réglage (présélection d'une temporisation ou d'un compteur, ajustement de la fréquence du générateur d'impulsions ou de la durée de préchauffage d'une machine, etc.).

Exemple

Utilisation du point de réglage analogique 1 pour modifier la durée de temporisation de 5 à 10 secondes :

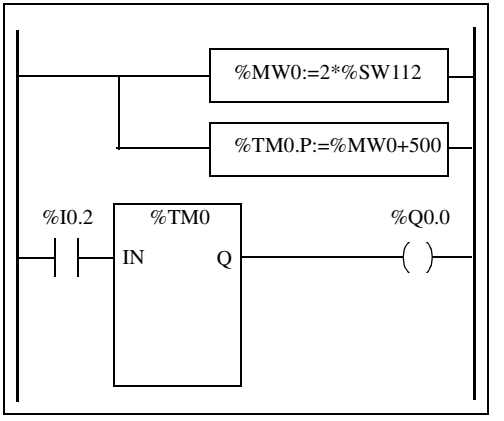
Ce réglage utilise la quasi-totalité de la plage du point de réglage analogique 1 (0 à 1023).	

Les paramètres suivants sont sélectionnés au moment de la configuration du bloc de temporisation %TM0 :

- Type TON
- Base temps (TB) : 10 ms

La valeur de présélection de la durée de temporisation est calculée à partir de la valeur du point de réglage analogique, à l'aide de l'équation suivante %TM0.P := 2*%SW112+500.

Code pour l'exemple précédent :




```
LD      1
[%MW0:=2*%SW112]
[%TM0.P:=%MW0+500]
BLK     %TM0
LD      %I0.0
IN
OUT_BLK
LD      Q
ST      %Q0.0
END_BLK
.....

END
```

Voie analogique

Introduction

Tous les automates modulaires (TWDLMDA20DTK, TWDLMDA20DUK, TWDLMDA20DRT, TWDLMD40DTK et TWDLMD40DUK) possèdent une voie analogique. La tension en entrée s'étend de 0 à 10 V et le signal numérisé de 0 à 511. La voie analogique utilise un schéma de calcul de moyennes simple qui s'applique sur huit échantillons.

Principe

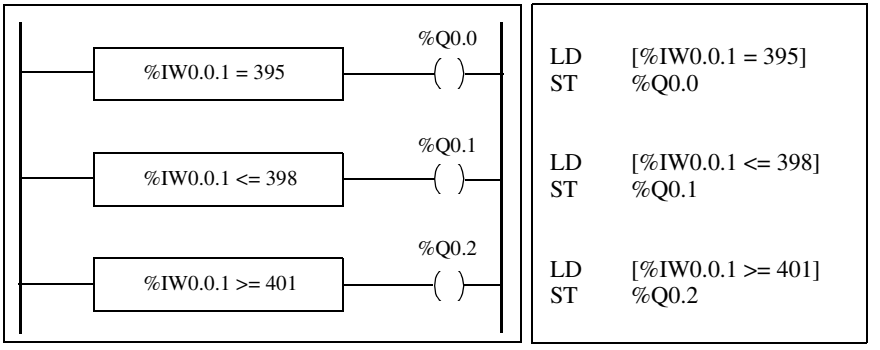
Un convertisseur de données analogiques en données numériques échantillonne une tension allant de 0 à 10 V en une valeur numérique allant de 0 à 511. Cette valeur est stockée dans le mot système %IW0.0.1. La valeur est linéaire sur l'intégralité de la plage, et chaque comptage est de 20 MV (10 V/512) approximativement. Une lecture de 511 est utilisée pour détecter le dépassement éventuel de la valeur maximale du signal en entrée.

Exemple de programmation :

Régulation de la température d'un four : La température de cuisson est réglée sur 350°C. Une variation de +/- 2,5°C engendre une disjonction des sorties %Q0.1 et %Q0.2. La quasi-totalité de la plage de paramètres possibles de la voie analogique (de 0 à 511) est utilisée dans cet exemple. Les paramètres analogiques des différentes températures sont les suivants :

Température (°C)	Tension	Mot système %IW0.0.1
0	0	0
347,5	7,72	395
350	7,77	398
352,5	7,83	401
450	10	511

Code de l'exemple précédent :



Gestion des modules analogiques

7

En bref...

Présentation

Ce chapitre offre une présentation des procédures de gestion des modules analogiques des automates Twido.

Contenu de ce chapitre

Ce chapitre contient les sujets suivants :

Présentation du module analogique

Introduction

Outre le point de réglage analogique 10 bits et la voie analogique 9 bits, l'ensemble des automates Twido prenant en charge l'expansion d'E/S sont également capables de configurer et de communiquer avec des modules d'E/S analogiques.
Ces modules analogiques sont les suivants :

Nom	Voies	Plage du signal	Codage
TWDAMI2HT	2 en entrée	0 - 10 V ou 4 - 20 mA	12 bits
TWDAM01HT	1 en sortie	0 - 10 V ou 4 - 20 mA	12 bits
TWDAMM3HT	2 en entrée, 1 en sortie	0 - 10 V ou 4 - 20 mA	12 bits
TWDALM3LT	2 en entrée, 1 en sortie	0 - 10 V, Entrées Th ou RTD, Sorties de 4 à 20 mA	12 bits

Fonctionnement des modules analogiques

Les mots en entrée et en sortie (%IW et %QW) sont utilisés pour échanger des données entre l'application utilisateur et les voies analogiques. La mise à jour de ces mots est effectuée de manière synchronisée avec la scrutation de l'automate en mode RUN.

ATTENTION

Mise en route non désirée d'équipements

Lorsque l'automate est en position STOP, la sortie analogique se trouve en position de repli. Dans le cas d'une sortie numérique, la position de repli est zéro.

Le non-respect de ces précautions peut entraîner des lésions corporelles ou/et des dommages matériels.

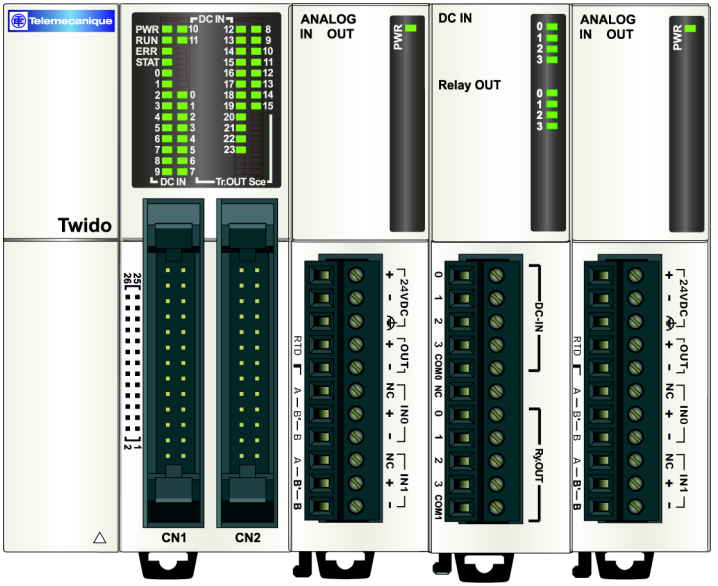
Adressage d’entrées et de sorties analogiques

Introduction

Des adresses sont affectées aux voies analogiques en fonction de leur emplacement sur le bus d’expansion.

Exemple d'adressage d'E/S analogique

Dans cet exemple, un module TWDLMDA40DUK possède un point de réglage analogique 10 bits intégré, ainsi qu'une voie analogique 9 bits intégrée. Sur le bus d'expansion, sont configurés : un module analogique TWDAMM3HT, un module de relais numérique d'E/S TWDDMM8DRT, ainsi qu'un second module analogique TWDAMM3HT.



Base Module 1 Module 2 Module 3

Le tableau suivant présente une description détaillée de l'adressage de chaque sortie.

Description	Base	Module 1	Module 2	Module 3
Pt de régl. analog. 1	%IW0.0.0			
Voie analogique intégrée ou Pt de régl. analog. 2	%IW0.0.1			
Voie 1 d'entrée analogique		%IW0.1.0		%IW0.3.0
Voie 2 d'entrée analogique		%IW0.1.1		%IW0.3.1
Voie 1 de sortie analogique		%QW0.1.0		%QW0.3.0
Voies d'entrée numérique			%I0.2.0 - %I0.2.3	
Voies de sortie numérique			%Q0.2.0 -%Q0.2.3	

Configuration d'entrées et de sorties analogiques

Introduction

Cette section présente des informations sur la configuration des entrées et des sorties du module analogique.

Configuration d'E/S analogique

La boîte de dialogue Configurer un module permet de gérer les paramètres des modules analogiques.

Note : Vous pouvez modifier ces paramètres en mode local, lorsque vous n'êtes pas connecté à un automate.

Des repères sont affectés aux voies analogiques en fonction de leur emplacement sur le bus d'expansion. Afin de vous aider dans vos travaux de programmation, vous pouvez également utiliser des symboles prédéfinis afin de faciliter la manipulation des données dans votre application utilisateur.
Vous pouvez configurer la voie de sortie unique de TWDAM01HT, TWDAMM3HT TWDALM3LT comme suit :

- Non utilisé
- 0 – 10 V
- 4 – 20 mA

Vous pouvez configurer les deux voies d'entrée de TWDAMI2HT et TWDAMM3HT comme suit :

- Non utilisé
- 0 – 10 V
- 4 – 20 mA

ATTENTION

Endommagement du matériel inattendu

Lorsque vous raccordez votre entrée afin de mesurer la tension et que vous configurez TwidoSoft pour un type courant de configuration, vous risquez d'endommager le module analogique de façon irréversible. Assurez-vous que le raccordement est conforme à la configuration TwidoSoft.

Le non-respect de ces précautions peut entraîner des lésions corporelles ou/et des dommages matériels.

Les deux voies d'entrée de TWDALM3LT peuvent être configurées comme suit :

- Non utilisé
- Thermocouple K
- Thermocouple J
- Thermocouple T
- PT 100

Lorsqu'une voie est configurée, vous pouvez lui affecter des unités et mapper la plage des entrées en fonction du tableau suivant.

Plage	**Unités**	**Description**
Normale	Aucune	Plage fixe allant de 0 à 4095 (valeurs minimale et maximale).
Personnalisée	Aucune	Définie par l'utilisateur, mais comprise entre –32 768 et +32 767.
Celsius	0.1°C	Echelle thermométrique internationale. Uniquement disponible pour les voies d'entrée TWDALM3LT.
Fahrenheit	0.1°F	Echelle thermométrique dans laquelle le point d'ébullition de l'eau est fixé à 212°F (100°C) et le point de gel à 32°F (0°C). Uniquement disponible pour les voies d'entrée TWDALM3LT.

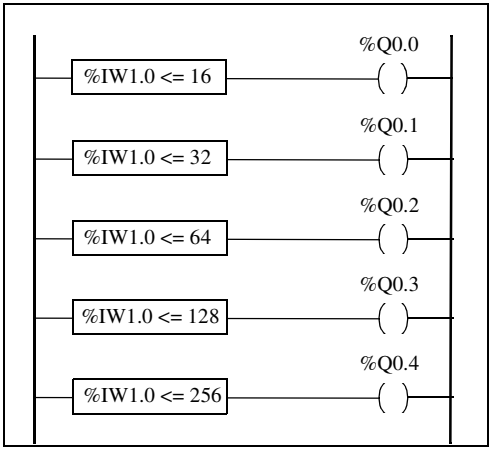
Exemples d'utilisation de modules analogiques

Introduction

Cette section présente un exemple d'utilisation des modules analogiques des automates Twido.

Exemple

Cet exemple compare le signal d'entrée analogique avec cinq valeurs de seuil distinctes. Une comparaison de l'entrée analogique est effectuée et un bit est réglé sur la base automate si le signal d'entrée est inférieur au seuil.




```
LD [%IW1.0 <= 16]
ST %Q0.0

LD [%IW1.0 <= 32]
ST %Q0.1

LD [%IW1.0 <= 64]
ST %Q0.2

LD [%IW1.0 <= 128]
ST %Q0.3

LD [%IW1.0 <= 256]
ST %Q0.4
```

Fonctionnement de l’afficheur

8

En bref...

Présentation

Ce chapitre offre des informations sur l’utilisation de l’afficheur optionnel Twido.

Contenu de ce chapitre

Ce chapitre contient les sujets suivants :

Afficheur

Introduction

L'afficheur est une option de Twido dont l'interface permet d'afficher et de contrôler les données de l'application et quelques fonctions de l'automate, telles que l'état de fonctionnement et l'horodateur (RTC). Cette option est disponible sous la forme d'une cartouche (TWDXCPODC) pour les automates compacts ou d'un module d'expansion (TWDXCPODM) pour les automates modulaires.
L'afficheur dispose de deux modes de fonctionnement :

- mode affichage : affiche simplement les données.
- mode édition : permet de modifier les données.

Note : L'afficheur est mis à jour selon un intervalle défini dans le cycle de scrutation de l'automate. Cela peut provoquer des erreurs d'interprétation de l'affichage des sorties dédiées pour les impulsions %PLS et %PWM. Au moment de l'échantillonnage de ces sorties, leur valeur sera toujours égale à zéro et c'est cette valeur qui sera affichée. Assurez-vous que la configuration du bloc fonction de régulation agit sur la sortie dédiée actuelle.

Affichage et fonctions

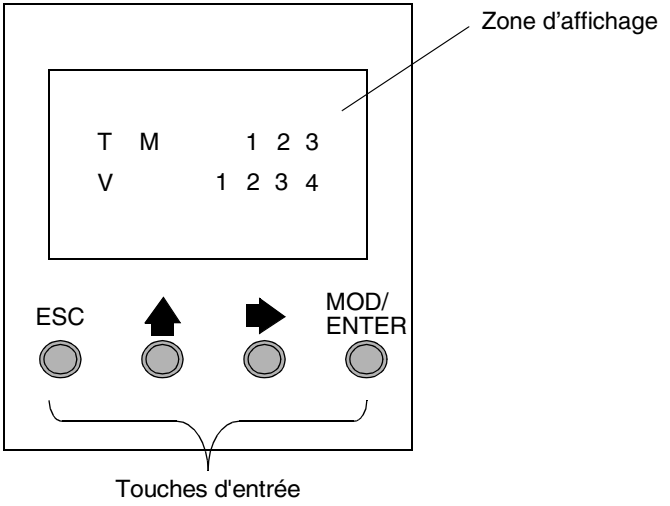
L'afficheur comporte différents affichages et les fonctions associées.

- informations sur l'identification et l'état de l'automate :
  affichage de la version du microprogramme et l'état de l'automate. Modification de l'état de l'automate à l'aide des commandes Run, Initial et Stop. Affichage des codes d'erreur à l'état Suspendu.
- variables et objets système :
  sélection des données de l'application par l'adresse : %I, %Q et tous les autres objets logiciels de la base automate. Contrôle et modification de la valeur de l'objet donnée logicielle sélectionné.
- paramètres du port série :
  affichage et configuration des paramètres du port de communication.
- horloge Date/Heure
  affichage et configuration de la date et de l'heure actuelles lorsque l'horodateur (RTC) est installé.
- facteur de correction du RTC :
  affichage et modification des valeurs de correction de l'horodateur (RTC) optionnel.

Note : L'horloge Date/Heure et la correction du RTC ne sont disponibles que lorsque la cartouche horodateur optionnelle (TWDXCPRTC) est installée.

Illustration

L'illustration suivante présente un schéma simplifié de l'afficheur, composé d'une zone d'affichage et de quatre touches d'entrée.



Zone d'affichage

L'afficheur est composé d'un écran à cristaux liquides capable d'afficher deux lignes de caractères.

- La première ligne de l'affichage est composée de trois caractères de 13 segments et de quatre caractères de 7 segments.
- La seconde ligne est composée d'un caractère de 13 segments, d'un caractère de 3 segments (pour les signes plus et moins) et de cinq caractères de 7 segments.

Touches d'entrée

Les fonctions des quatre touches d'entrée dépendent du mode de l'afficheur.

Touche	En mode affichage	En mode édition
ESC		Annule les modifications et revient à l'affichage précédent.
▲		Attribue la valeur de l'élément suivant à l'élément en cours d'édition.
➡	Passe à l'affichage suivant.	Passe à l'élément à éditer suivant.
MOD/ ENTER	Passe en mode édition.	Accepte les modifications et revient à l'affichage précédent.

Sélection et navigation entre les affichages

L'affichage ou l’écran initial de l'afficheur présente des informations sur l'identification et l'état de l'automate. Appuyez sur la touche ➡ pour passer d'un affichage à l'autre. Les écrans de l'horloge Date/Heure et le facteur de correction du RTC apparaissent uniquement lorsque la cartouche horodateur (TWDXCPRTC) est détectée sur l'automate.

Appuyez sur la touche ESC pour revenir à l'écran initial. Pour la plupart des écrans, le fait de libérer la touche ESC permet de revenir à l'écran Informations d'identification et états de l'automate. Le fait d'appuyer sur la touche ESC permet uniquement de retourner à la saisie du premier objet système ou de l'objet système initial lors de la modification de variables et d’objets système autres que l’entrée initiale (%I0.0.0).

Pour modifier la valeur d'un objet, appuyez à nouveau sur la touche MOD/ENTER au lieu d'appuyer sur la touche ➡ pour accéder au premier chiffre de la valeur.

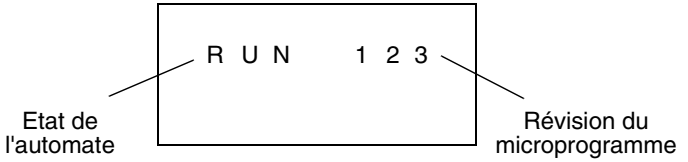
Informations d'identification et états de l'automate

Introduction

L'écran initial de l'afficheur optionnel Twido présente des informations sur l'identification et sur l'état de l'automate.

Exemple

Comme l'illustre le schéma suivant, la version du microprogramme est affichée dans le coin supérieur droit de la zone d'affichage, l'état de l'automate dans le coin supérieur gauche.



Etats de l’automate

L'automate peut se trouver dans l'un des états suivants :

- **NCF : Non configuré**
  L'automate demeure en état NCF jusqu'à ce qu'une application soit chargée. Aucun autre état n'est permis avant le chargement du programme de l'application. Vous pouvez tester l'E/S en modifiant le bit système S8 (reportez-vous à la rubrique *Bits système (%S), p. 332*).
- **STP : Arrêté**
  Dès qu'une application est chargée sur l'automate, ce dernier passe à l'état STP. Dans cet état, l'application ne fonctionne pas. Les entrées sont mises à jour et les valeurs des données restent inchangées. Les sorties ne sont pas mises à jour dans cet état.
- **INI : Initial**
  Seul un automate se trouvant à l'état STP peut passer à l'état INI. L'application n'est pas en cours d'exécution. Les entrées de l'automate sont mises à jour et les valeurs des données retrouvent leur état initial. Aucune sortie n'est mise à jour dans cet état.
- **RUN : En cours d'exécution**
  Dans cet état, l'application fonctionne. Les entrées de l'automate sont mises à jour et les valeurs des données sont réglées par l'application. Il s'agit du seul état au cours duquel les sorties sont mises à jour.
- **HLT : Suspendu (Erreur d’application utilisateur)**
  L'exécution de l'application est suspendue dès que l'automate passe à l'état ERR. Les entrées sont mises à jour et les valeurs des données restent inchangées. Dans cet état, les sorties ne sont pas mises à jour. Dans ce mode, le code de l'erreur est affiché dans la partie inférieure droite de l'afficheur. Ce code prend la forme d'une valeur décimale sans signe.
- **NEX : Not Executable (non exécutable)**
  Une modification en ligne a été apporté e à la logique utilisateur. Conséquence : l'application n'est plus exécutable. Elle ne retrouvera cet état qu'une fois que toutes les causes de l'état Non Exec auront été résolves.

Affichage et modification des états de l'automate

L'afficheur vous permet de faire passer l'automate de l'état STP à l'état INI, de l'état STP à l'état RUN, ou de l'état RUN à l'état STP. Pour modifier l'état de l'automate, procédez comme suit :

Etape	Action
1	Appuyez sur la touche ➡ jusqu'à ce que l'écran Affichage des opérations apparaisse (ou appuyez sur la touche ESC). L'état courant de l'automate apparaît dans le coin supérieur gauche de la zone d'affichage.
2	Appuyez sur la touche MOD/ENTER pour passer en mode édition.
3	Appuyez sur la touche ⬆ pour sélectionner un état de l'automate.
4	Appuyez sur la touche MOD/ENTER pour accepter la valeur modifiée, ou sur la touche ESC pour ignorer les modifications apportées en mode édition.

Variables et objets système

Introduction

L'Afficheur optionnel permet de contrôler et d'ajuster les données de l'application à l'aide des fonctions suivantes :

- sélection des données de l'application par l'adresse (telle que %I ou %Q) ;
- contrôle de la valeur de l'objet/variable logiciel(le) sélectionné(e) ;
- modification de la valeur de l'objet donnée actuellement affiché (y compris le forçage des entrées et des sorties).

Variables et objets système

Le tableau ci-après répertorie, dans leur ordre d'accès, les variables et objets système qui peuvent être affichés et modifiés par l'Afficheur.

Objet	Variable/Attribut	Description	Accès
Entrée	%I.x.y.z	Valeur	Lecture/Forçage
Sortie	%Q.x.y.z	Valeur	Lecture/Ecriture/ Forçage
Temporisateur	%TMX.V %TMX.P %TMX.Q	Valeur courante Valeur de présélection Terminé	Lecture/Ecriture Lecture/Ecriture Lecture
Compteur	%Cx.V %Cx.P %Cx.D %Cx.E %Cx.F	Valeur courante Valeur de présélection Terminé Vide Plein	Lecture/Ecriture Lecture/Ecriture Lecture Lecture Lecture
Bit mémoire	%Mx	Valeur	Lecture/Ecriture
Mot mémoire	%MWx	Valeur	Lecture/Ecriture
Mot constante	%KWx	Valeur	Lecture
Bit système	%Sx	Valeur	Lecture/Ecriture
Mot système	%SWx	Valeur	Lecture/Ecriture
Entrée analogique	%IW.x.y.z	Valeur	Lecture
Sortie analogique	%QW.x.y.z	Valeur	Lecture/Ecriture
Compteur rapide (FC)	%FCx.V %FCx.P %FCx.D	Valeur courante Valeur de présélection Terminé	Lecture/Ecriture Lecture/Ecriture Lecture

Objet	Variable/Attribut	Description	Accès
Compteur rapide (VFC)	%VFCx.V %VFCx.P %VFCx.U %VFCx.C %VFCx.S0 %VFCx.S1 %VFCx.F %VFCx.M %VFC.T %VFC.R %VFC.S	Valeur courante Valeur de présélection Sens de comptage Valeur de capture Valeur de seuil 0 Valeur de seuil 1 Débordement Fréquence mesurée Base temps Activation sortie réflexe Activation entrée réflexe	Lecture/Ecriture Lecture/Ecriture Lecture Lecture Lecture/Ecriture Lecture/Ecriture Lecture Lecture/Ecriture Lecture/Ecriture Lecture/Ecriture Lecture/Ecriture
Entrée mot réseau	%INWx.z	Valeur	Lecture/Ecriture
Sortie mot réseau	%QNWx.z	Valeur	Lecture/Ecriture
Grafcet	%Xx	Bit étape	Lecture
Générateur d'impulsions	%PLS.N %PLS.P %PLS.D %PLS.Q	Nombre de pulsations Valeur de présélection Terminé Sortie courante	Lecture/Ecriture Lecture/Ecriture Lecture Lecture
Modulateur de largeur d'impulsion	%PMW.R %PMW.P	Ratio Valeur de présélection	Lecture/Ecriture Lecture/Ecriture
Programmateur cyclique	%DRx.S %DRx.F	Nombre d'étapes courantes Plein	Lecture Lecture
Fonction pas à pas	%SCx.n	Bit de fonction pas à pas	Lecture/Ecriture
Registre	%Rx.I %Rx.O %Rx.E %Rx.F	Entrée Sortie Vide Plein	Lecture/Ecriture Lecture Lecture Lecture
Registre bits à décalage	%SBR.x.yy	Bit de registre	Lecture/Ecriture
Message	%MSGx.D %MSGx.E	Terminé Erreur	Lecture Lecture

Remarques :

1. Les variables n'apparaîtront pas si elles ne sont pas utilisées dans une application, étant donné que Twido utilise l'affectation de mémoire dynamique.
2. Si la valeur de %MW est supérieure à +32767 ou inférieure à -32787, l'afficheur continue de clignoter.
3. Si la valeur de %SW est supérieure à 65 535, l'afficheur continue de clignoter, sauf pour %SW0 et pour %SW11. Lorsqu'une valeur dépassant les limites est entrée, elle est remplacée par la valeur configurée.
4. Lorsqu'une valeur dépassant les limites est entrée pour %PLS.P, la valeur est définie sur la saturation.

Affichage et modification des objets et des variables

Vous pouvez accéder à chaque type d'objet système en commençant par l'objet entrée (%I), en progressant de façon séquentielle jusqu'à l'objet message (%MSG) et en revenant finalement à l'objet entrée (%I).
Pour afficher un objet système :

Etape	Action
1	Appuyez sur la touche ➡ jusqu'à ce que l'écran Affichage des données apparaisse. L'objet Entrée (« I ») apparaît dans le coin supérieur gauche de la zone d'affichage. La lettre « I » (ou le nom de l'objet précédent) ne clignote pas.
2	Appuyez sur la touche MOD/ENTER pour activer le mode édition. La lettre « I » de l'objet Entrée (ou le nom de l'objet précédent) commence à clignoter.
3	Appuyez sur la touche ⬆ pour progresser de façon séquentielle dans la liste des objets.
4	Appuyez sur la touche ➡ pour progresser de façon séquentielle dans le champ d'un type d'objet et appuyez sur la touche ⬆ pour incrémenter la valeur de ce champ. Utilisez les touches ➡ et ⬆ pour consulter et modifier tous les champs de l'objet affiché.
5	Répétez les étapes 3 et 4 jusqu'à ce que l'édition soit terminée.
6	Appuyez sur la touche MOD/ENTER pour accepter les valeurs modifiées. Remarque : Le nom et l'adresse de l'objet doivent être validés pour pouvoir accepter ces modifications. Cela signifie qu'ils doivent exister dans la configuration de l'automate avant l'utilisation de l'afficheur. Appuyez sur la touche ESC pour annuler les modifications apportées en mode édition.

Valeurs des données et formats d'affichage

En général, la valeur de donnée pour un objet ou une variable est affichée comme un entier avec signe ou sans signe dans la partie inférieure droite de la zone d'affichage. Les zéros non significatifs sont supprimés de tous les champs pour l'affichage des valeurs. L'adresse de chaque objet apparaît dans l'Afficheur dans l'un des six formats suivants :

- format E/S
- format bloc fonction
- format simple
- format E/S réseau
- format fonction pas à pas
- format registre bits à décalage.

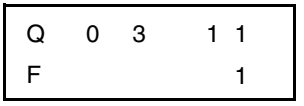
Format Entrée/ Sortie

Les objets entrée/sortie (%I, %Q, %IW et %QW) ont une adresse en trois parties (ex : %IX.Y.Z) et apparaissent sous la forme suivante :

- type d'objet et adresse de l'automate dans la partie supérieure gauche ;
- adresse de l'expansion dans la partie supérieure centrale ;
- voie d'E/S dans la partie supérieure droite.

Dans le cas d'une entrée (%I) et d'une sortie (%Q) simples, la lettre « U » pour un bit non forcé (unforced) ou la lettre « F » pour un bit forcé (forced) apparaît dans la partie inférieure gauche de l'affichage. La valeur de forçage apparaît dans la partie inférieure droite de l'écran.

L'objet sortie %Q0.3.11 apparaît dans la zone d'affichage sous la forme suivante :

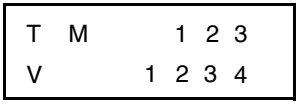


Format bloc fonction

Les blocs fonctions (%TM, %C, %FC, %VFC, %PLS, %PWM, %DR, %R et %MSGj) ont une adresse en deux parties comprenant le numéro de l'objet et le nom d'une variable ou d'un attribut. Ils apparaissent sous la forme suivante :

- nom du bloc fonction en haut à gauche ;
- numéro (ou instance) du bloc fonction en haut à droite ;
- variable ou attribut en bas à gauche ;
- valeur de l'attribut en bas à droite.

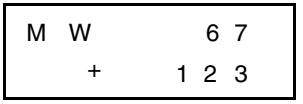
Dans l'exemple suivant, la valeur courante pour le temporisateur numéro 123 est réglée sur 1 234.



Format simple

Un format simple est utilisé pour les objets %M, %MW, %KW, %S, %SW et %X :
- numéro de l'objet dans la partie supérieure droite ;
- valeur avec signe pour les objets dans la partie inférieure.

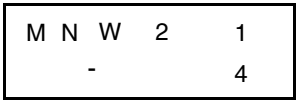
Dans l'exemple suivant, le mot mémoire numéro 67 contient la valeur +123.



Format Entrée/ Sortie réseau

Les objets entrée/sortie réseau (%INW et %QNW) apparaissent dans la zone d'affichage sous la forme suivante :
- nom de l'objet dans la partie supérieure gauche ;
- adresse de l'automate dans la partie supérieure centrale ;
- numéro de l'objet dans la partie supérieure droite ;
- Valeur avec signe de l'objet dans la partie inférieure.

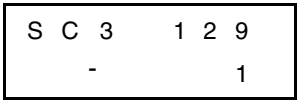
Dans l'exemple suivant, le premier mot d'entrée ou mot réseau de l'automate distant configuré à l'adresse distante n°2 a pour valeur -4.



Format fonction pas à pas

Le format fonction pas à pas (%SC) affiche le numéro de l'objet et le bit de fonction pas à pas sous la forme suivante :
- nom et numéro de l'objet dans la partie supérieure gauche ;
- bit de fonction pas à pas dans la partie supérieure droite ;
- valeur de l'objet dans la partie inférieure.

Dans l'exemple suivant, le bit numéro 129 de la fonction pas à pas numéro 3 est réglé sur -1.

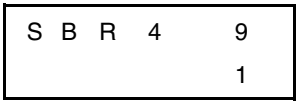


Format registre bits à décalage

Le format registre bit à décalage (%SBR) affiche le numéro de l'objet et le bit registre sous la forme suivante :

- nom et numéro de l'objet dans la partie supérieure gauche ;
- bit de registre dans la partie supérieure droite.

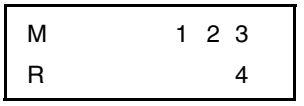
Vous trouverez ci-après un exemple de l'affichage du registre bit à décalage numéro 4.



Paramètres de port série

Introduction

L'afficheur vous permet de visualiser et de modifier les paramètres d'un protocole. Un maximum de deux ports série peuvent être utilisés. Dans l'exemple suivant, le premier port est configuré pour le protocole Modbus et porte l'adresse 123. Le second port est configuré en tant que liaison distante et porte l'adresse 5.



Affichage et modification des paramètres d'un port série

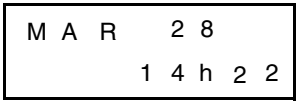
Les automates Twido peuvent gérer un maximum de deux ports série. Pour visualiser les paramètres des ports série sur l'afficheur :

Etape	Action
1	Appuyez sur la touche ➡ jusqu'à ce que l'écran Affichage des communications apparaisse. Une lettre, correspondant au paramètre de protocole du premier port (M, R ou A), sera affichée dans le coin supérieur gauche de l'afficheur.
2	Appuyez sur la touche MOD/ENTER pour passer en mode édition.
3	Appuyez sur la touche ➡ jusqu'à ce que vous vous trouviez dans le champ à modifier.
4	Appuyez sur la touche ⬆ pour incrémenter la valeur de ce champ.
5	Répétez les étapes 3 et 4 jusqu'à ce que tous les paramètres du port série aient été définis.
6	Appuyez sur la touche MOD/ENTER pour enregistrer les modifications apportées en mode édition ou sur ESC pour les ignorer.

Horloge Date/Heure

Introduction

Les paramètres de date et d'heure ne peuvent être mis à jour depuis l'afficheur que si la cartouche optionnelle de l'horodateur (TWDXCPRTC) est installée dans votre automate Twido. Le mois apparaît dans le coin supérieur gauche de l'afficheur. La valeur « RTC » figurera dans ce champ jusqu'à ce que des paramètres de date et d'heure valides aient été entrés. Le jour du mois apparaît dans le coin supérieur droit de l'afficheur. Cette heure est affichée au format dit « militaire ». Les heures et les minutes sont affichées dans le coin inférieur droit de l'afficheur et sont séparées par la lettre « h ». L'exemple suivant illustre ce qu'indiquerait l'afficheur, le 28 mars à 14:22.



Affichage et modification de l'horloge Date/ Heure

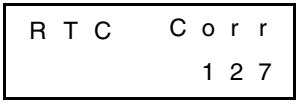
Pour afficher et modifier l'horloge Date/Heure :

Etape	Action
1	Appuyez sur la touche ➡ jusqu'à ce que l'écran Affichage Date/Heure apparaisse. Le code du mois (« JAN » ou « FEV », par exemple) apparaît dans le coin supérieur gauche de la zone d'affichage. La mention « RTC » est affichée tant que le mois n'a pas été défini.
2	Appuyez sur la touche MOD/ENTER pour passer en mode édition.
3	Appuyez sur la touche ➡ jusqu'à ce que vous vous trouviez dans le champ à modifier.
4	Appuyez sur la touche ⬆ pour incrémenter la valeur de ce champ.
5	Répétez les étapes 3 et 4 jusqu'à ce que tous les paramètres de date et d'heure aient été définis.
6	Appuyez sur la touche MOD/ENTER pour enregistrer les modifications apportées en mode édition ou sur ESC pour les ignorer.

Facteur de correction de l'horodateur

Introduction

L'afficheur vous permet de visualiser et de modifier le facteur de correction de l'horodateur (RTC). Pour chaque module option horodateur (RTC), une valeur de correction du RTC permet de corriger les imprécisions du cristal du module RTC. Ce facteur prend la forme d'un nombre entier sans signe, composé de trois chiffres, compris entre 0 et 127. Cette valeur apparaît dans le coin supérieur droit de l'afficheur.
L'exemple suivant illustre un facteur de correction de 127.



Affichage et modification de la correction du RTC

Pour afficher et modifier le facteur de correction du RTC :

Etape	Action
1	Appuyez sur la touche ➡ jusqu'à ce que l'écran Affichage du facteur RTC apparaisse. "RTC Corr" s'affiche dans la ligne supérieure de l'afficheur.
2	Appuyez sur la touche MOD/ENTER pour passer en mode édition.
3	Appuyez sur la touche ➡ jusqu'à ce que vous vous trouviez dans le champ à modifier.
4	Appuyez sur la touche ⬆ pour incrémenter la valeur de ce champ.
5	Répétez les étapes 3 et 4 jusqu'à ce que la valeur de correction du RTC ait été définie.
6	Appuyez sur la touche MOD/ENTER pour enregistrer les modifications apportées en mode édition ou sur ESC pour les ignorer.

Description des langages Twido

III

En bref...

Présentation

Cette rubrique fournit des instructions d'utilisation des langages de programmation Grafcet, schéma à contacts et liste d'instructions permettant de créer des programmes de régulation des automates programmables Twido.

Contenu de cet intercalaire

Cet intercalaire contient les chapitres suivants :

Langage schéma à contacts

9

En bref...

Présentation

Cette rubrique décrit la programmation à l'aide du langage schéma à contacts.

Contenu de ce chapitre

Ce chapitre contient les sujets suivants :

Introduction aux schémas à contacts

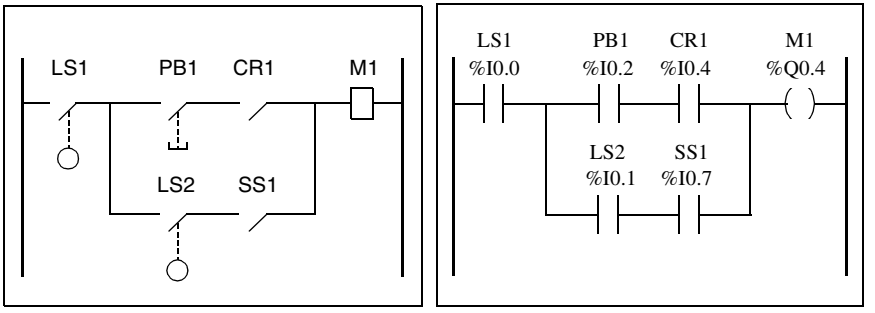
Introduction

Les schémas à contacts utilisent la même représentation graphique que celle des circuits de relais en logique programmée, à ceci près que, dans un schéma à contacts :

- Toutes les entrées sont représentées par des symboles de contacts (⊣⊢).
- Toutes les sorties sont représentées par des symboles de bobines (⊣( )⊢).
- Les opérations numériques sont comprises dans le jeu d'instructions graphiques du schéma à contacts.

Représentations de schémas à contacts correspondant aux circuits de relais

L'illustration suivante présente un schéma simplifié de câblage d'un relais en logique programmée, et son équivalent en langage schéma à contacts.



Circuit de relais en logique programmée — Schéma à contacts

Dans l'illustration précédente, toutes les entrées associées à un périphérique de commutation dans le circuit de relais en logique programmée sont représentées sous la forme de contacts dans le schéma à contacts. La bobine de sortie M1 du circuit logique de relais est représentée par un symbole de bobine dans le schéma à contacts. Les numéros des adresses apparaissant au-dessus du symbole de chaque contact et de chaque bobine dans le schéma à contacts sont des références aux emplacements des connexions externes en entrée et en sortie vers l'automate.

Réseaux schéma à contacts

Un programme en langage schéma à contacts est composé de « réseaux », représentant des ensembles d'instructions graphiques et apparaissant entre deux barres verticales. Les réseaux sont exécutés de manière séquentielle par l'automate.

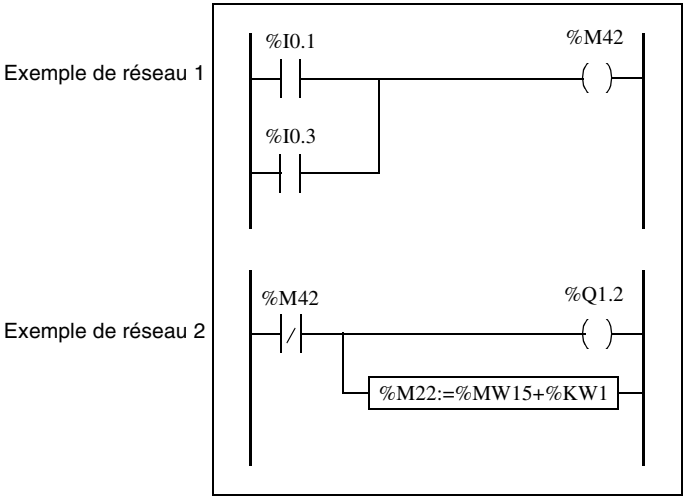
L'ensemble des instructions graphiques représente les fonctions suivantes :

- entrées/sorties de l'automate (boutons de commande, capteurs, relais, voyants, etc.)
- fonctions de l'automate (temporisateurs, compteurs, ...)
- opérations mathématiques et logiques (addition, division, AND, XOR, etc.)
- opérateurs de comparaison et autres opérations numériques (A<B, A=B, décalage, rotation, etc.)
- variables internes de l'automate (bits, mots, etc.)

Ces instructions sont disposées graphiquement à l'aide de barres verticales et horizontales et peuvent déboucher sur une ou plusieurs sorties et/ou actions. Un réseau ne peut pas contenir plus d'un groupe d'instructions liées.

Exemple de réseaux schéma à contacts

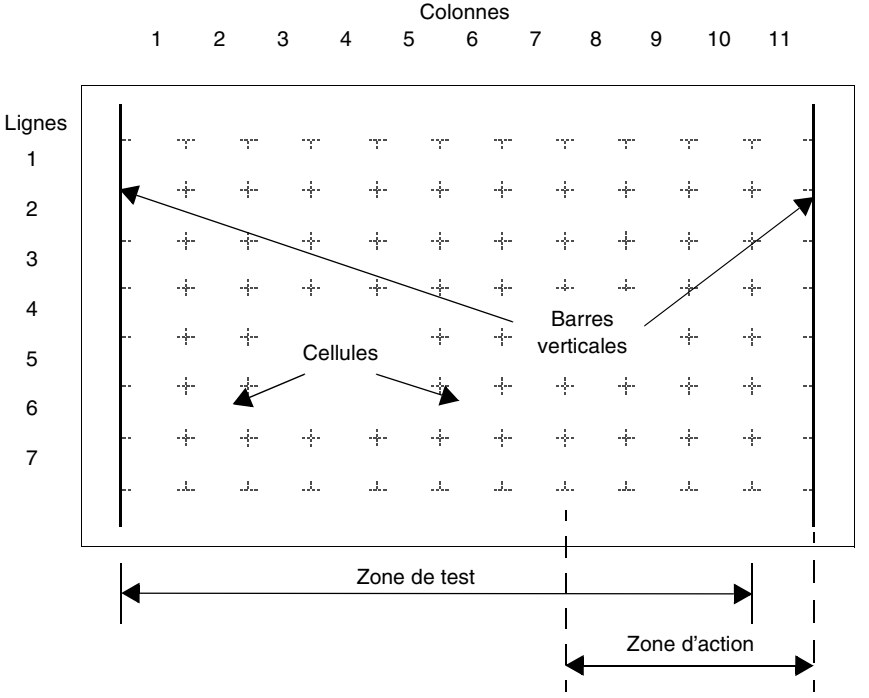
L'exemple suivant illustre un programme en langage schéma à contacts composé de deux réseaux.



Principes de programmation en langage schéma à contacts

Grille de programmation

Chaque réseau schéma à contacts se compose d'une grille comportant sept lignes et onze colonnes organisées en deux zones, comme l'indique l'illustration suivante :



Zones de la grille

La grille de programmation en langage schéma à contacts est divisée en deux zones :

- Zone de test
  Contient les conditions testées avant d'effectuer des actions. Comprend les colonnes 1 à 10 et contient les contacts, les blocs fonctions et les blocs comparaisons.
- Zone d'action
  Contient la sortie ou l'opération qui sera effectuée en fonction des résultats des tests réalisés sur les conditions dans la zone de test. Comprend les colonnes 8 à 11 et contient les bobines et les blocs opérations.

Saisie d'instructions dans la grille

La grille de sept lignes sur onze colonnes que constitue le réseau schéma à contacts se lit à partir de la cellule située en haut à gauche. La programmation consiste à entrer des instructions dans les cellules de la grille. Les instructions de test, de comparaison et de fonctions sont entrées dans les cellules de la zone de test et sont justifiées à gauche. La logique du test permet d'assurer la continuité dans la zone d'action, où les bobines, les opérations numériques et les instructions de régulation du flux du programme sont entrées et justifiées à droite.
Le réseau est traité ou exécuté (tests effectués et sorties affectées) dans la grille de haut en bas et de gauche à droite.

En-tête réseau

Un en-tête apparaît directement au-dessus du réseau. Vous pouvez l'utiliser pour donner des informations sur la finalité logique du réseau. L'en-tête de réseau peut contenir les informations suivantes :

- le numéro du réseau ;
- des étiquettes (%Li) ;
- des déclarations de sous-programme (SRi:) ;
- le titre du réseau ;
- des commentaires sur le réseau.

Pour obtenir davantage d'informations sur l'utilisation d'un en-tête réseau pour documenter vos programmes, reportez-vous à la rubrique *Documentation du programme, p. 173*.

Blocs de schémas à contacts

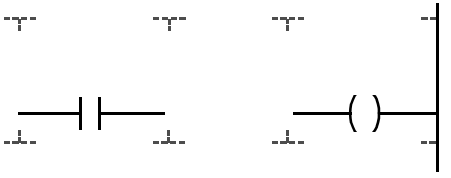
Introduction

Les schémas à contacts se composent de blocs correspondant à des actions et/ou des fonctions d'un programme, telles que :

- des contacts
- des bobines
- des instructions de déroulement du programme
- des blocs fonctions
- des blocs comparaisons
- des blocs opérations

Contacts, bobines et déroulement du programme

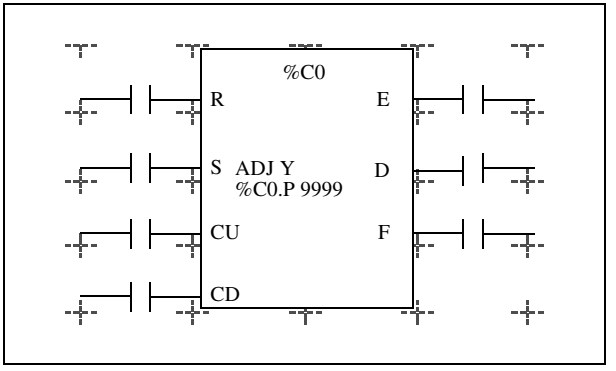
Les contacts, bobines et les instructions de déroulement du programme (sauts et appels) n'occupent qu'une seule cellule dans la grille de programmation du schéma à contacts. Les blocs fonctions, les blocs comparaisons et les blocs opérations peuvent en revanche occuper plusieurs cellules.
Les exemples suivants illustrent un contact et une bobine.



Contact Bobine

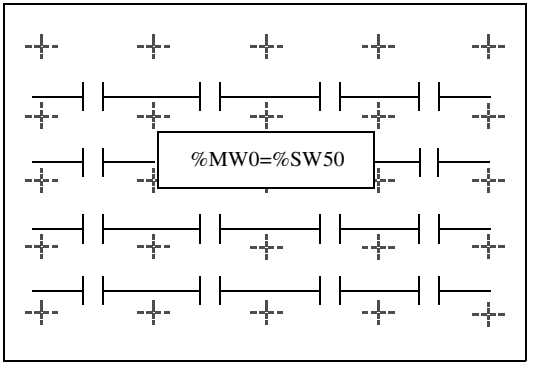
Blocs fonctions

Les blocs fonctions sont placés dans la zone de test de la grille de programmation. Le bloc doit figurer sur la première ligne ; aucune instruction de schéma à contacts ou aucune ligne de continuité ne peut apparaître au-dessus ou en dessous du bloc fonction. Les instructions de test du schéma à contacts mènent à l'entrée du bloc fonction, alors que les instructions de test et/ou les instructions d'action proviennent de la sortie du bloc.
Les blocs fonctions sont orientés de manière verticale et occupent deux colonnes sur quatre lignes dans la grille de programmation.
L'exemple suivant illustre un bloc fonction temporisateur.



Blocs comparaisons

Les blocs comparaisons sont placés dans la zone de test de la grille de programmation. Le bloc peut apparaître sur n'importe quelle ligne ou colonne de la zone de test. L'intégralité de l'instruction doit résider dans cette zone.
Les blocs comparaisons sont orientés de manière horizontale et occupent deux colonnes sur une ligne dans la grille de programmation.
L'exemple suivant présente un bloc comparaison.

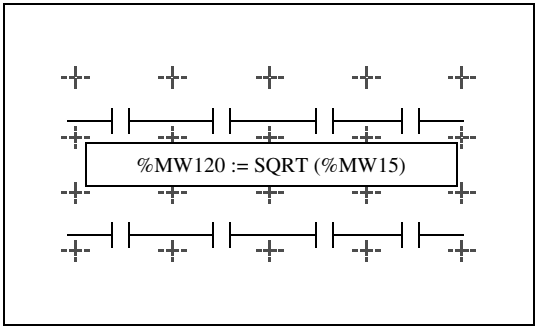


Blocs opérations

Les blocs opérations sont placés dans la zone d'action de la grille de programmation. Le bloc peut apparaître sur n'importe quelle ligne de la zone d'action. L'instruction est justifiée à droite ; elle apparaît à droite et se termine dans la dernière colonne.

Les blocs opérations sont orientés de manière horizontale et occupent quatre colonnes sur une ligne dans la grille de programmation.

L'exemple suivant illustre un bloc opération.



éléments graphiques du langage schéma à contacts

Introduction

Les instructions des schémas à contacts sont constituées d'éléments graphiques. Cette section répertorie et décrit les éléments graphiques utilisés dans les instructions du langage schéma à contacts Twido. Reportez-vous au guide d'exploitation TwidoSoft pour plus de détails sur l'utilisation de ces éléments graphiques dans les programmes par schémas à contacts Twido.

Contacts

Les éléments graphiques des contacts sont programmés dans la zone de test et occupent une cellule (une ligne sur une colonne).

Nom	Elément graphique	Instruction	Fonction
Contact à ouverture		LD	Fait passer le contact lorsque l'objet bit de régulation se trouve à l'état 1.
Contact à fermeture		LDN	Fait passer le contact lorsque l'objet bit de régulation se trouve à l'état 0.
Contact de détection d'un front montant		LDR	Front montant : détecte le passage de 0 à 1 de l'objet bit de régulation.
Contact de détection d'un front descendant		LDF	Front descendant : détecte le passage de 1 à 0 de l'objet bit de régulation.

Eléments de liaison

Les éléments de liaison graphique s'utilisent pour connecter les éléments graphiques de test et d'action.

Nom	Elément graphique	Fonction
Connecteur horizontal		Relie en série les éléments graphiques de test et d'action entre les deux barres verticales.
Connecteur secondaire		Relie les éléments graphiques de test et d'action en parallèle (connexion verticale).

Bobines

Les éléments graphiques des bobines sont programmés dans la zone d'action et occupent une cellule (une ligne sur une colonne).

Nom	Elément graphique	Instruction	Fonction
Bobine directe		ST	L'objet bit associé prend la valeur du résultat de la zone de test.
Bobine négation		STN	L'objet bit associé prend la valeur du résultat négatif de la zone de test.
Bobine SET		S	L'objet bit associé est réglé sur 1 lorsque le résultat de la zone de test est 1.
Bobine RESET		R	L'objet bit associé est réglé sur 0 lorsque le résultat de la zone de test est 1.
Appel de saut ou de sous-programme	->>%Li ->>%SRi	JMP SR	Se connecte à une instruction portant une étiquette, en amont ou en aval.
Bobine dièse			Langage Grafcet. Utilisée lorsque la programmation des conditions de transition associées aux transitions provoque une permutation sur l'étape suivante.
Retour d'un sous-programme	<RET>	RET	Placé à la fin des sous-programmes pour retourner au programme principal.
Arrêt du programme	<END>	END	Définit la fin du programme.

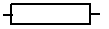
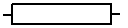
Blocs fonctions

Les éléments graphiques des blocs fonctions sont programmés dans la zone de test et occupent quatre lignes sur deux colonnes (excepté les compteurs rapides (VFC), qui requièrent cinq lignes sur deux colonnes).

Nom	Elément graphique	Fonction
Temporisateurs, compteurs, registres, etc.		Chaque bloc fonction utilise les entrées et les sorties permettant la liaison aux autres éléments graphiques. Note : Les sorties des blocs fonctions ne peuvent pas être connectées les unes aux autres (liaisons verticales).

Blocs opérations et comparaisons

Les blocs comparaisons sont programmés dans la zone de test et les blocs opérations, dans la zone d'action.

Nom	Elément graphique	Fonction
Bloc comparaison		Compare deux opérandes. La sortie prend la valeur 1 lorsque le résultat est vérifié. Taille : Une ligne sur deux colonnes
Bloc opération		Effectue des opérations arithmétiques et logiques. Taille : Une ligne sur quatre colonnes

Instructions spéciales OPEN et SHORT du langage schéma à contacts

Introduction

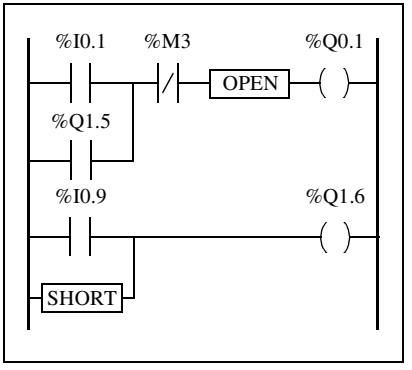
Les instructions OPEN et SHORT permettent de déboguer rapidement et simplement des programmes en langage schéma à contacts. Ces instructions spéciales modifient la logique d'un réseau, soit en raccourcissant, soit en ouvrant la continuité d'un réseau, conformément aux explications fournies dans le tableau suivant.

Instruction	Description	Instruction en langage liste d'instructions
OPEN	Crée un arrêt dans la continuité d'un réseau schéma à contacts, et ce, quels que soient les résultats de la dernière opération logique.	`AND 0`
SHORT	Permet à la continuité de traverser le réseau schéma à contacts, et ce, quels que soient les résultats de la dernière opération logique.	`OR 1`

En langage liste d'instructions, les instructions OR et AND sont utilisées pour créer les instructions OPEN et SHORT à l'aide des valeurs immédiates respectives de 0 et 1.

Exemples

Les exemples suivants illustrent l'utilisation des instructions SHORT et OPEN.



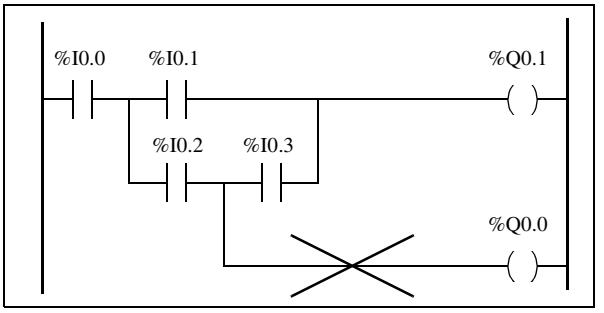

```
LD      %I0.1
OR      %Q1.5
ANDN    %M3
AND     0
ST      %Q0.1
LD      %I0.9
OR      1
ST      %Q1.6
```

Conseils de programmation

Gestion de sauts de programme	Utilisez les sauts de programme avec la plus grande précaution, car ils peuvent être à l'origine de boucles qui ralentiront considérablement les opérations de scrutation. Evitez d'insérer des sauts pointant vers des instructions situées en amont. Une instruction en amont apparaît avant un saut dans un programme. A l'inverse, une instruction en aval apparaît après un saut dans un programme.
Programmation des sorties	Les bits de sortie, tout comme les bits internes, ne peuvent être régulés que s'ils figurent dans le programme. Pour les bits de sortie, seule la dernière valeur scrutée est prise en compte lors de la mise à jour des sorties.
Utilisation de capteurs d'arrêt d'urgence à liaison directe	Les capteurs utilisés en cas d'arrêt d'urgence ne doivent pas être gérés par l'automate. Ces capteurs doivent être raccordés directement aux sorties correspondantes.
Gestion des reprises de l'alimentation	Lors d'un fonctionnement manuel, les reprises de l'alimentation doivent être conditionnelles. En effet, un redémarrage automatique de l'installation pourrait provoquer une mise sous tension non souhaitée de certains équipements (utilisez les bits système %S0, %S1 et %S9).
Gestion des blocs Heure et Horodateur	Il est nécessaire de vérifier l'état du bit système %S51, qui indique d'éventuels défauts de bloc Horodateur.
Vérification de la syntaxe et recherche d'erreurs	Lors de la saisie d'un programme, TwidoSoft vérifie la syntaxe de ses instructions et opérandes, ainsi que leur association. Pour plus d'informations, reportez-vous au guide d'exploitation TwidoSoft.

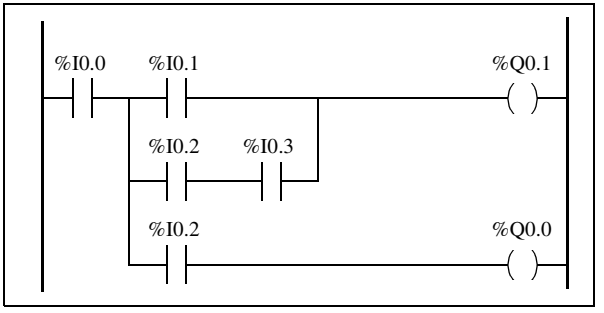
Remarques complémentaires sur l'utilisation des parenthèses

Les opérations d'affectation ne doivent pas être placées entre parenthèses :



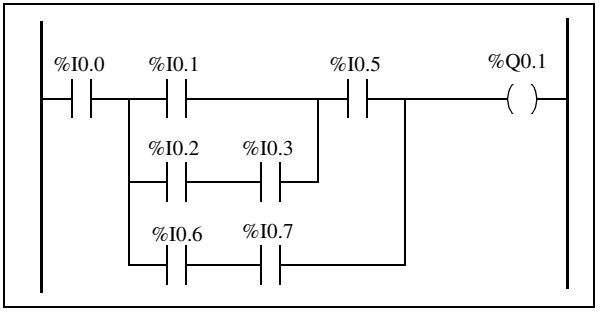

```
LD      %I0.0
AND     %I0.1
OR(     %I0.2
ST      %Q0.0
AND     %I0.3
)
ST      %Q0.1
```

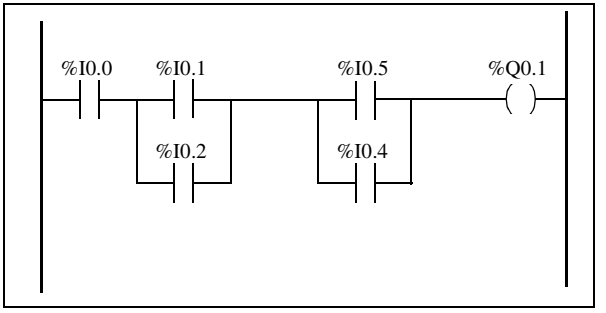
Afin d'effectuer la fonction correspondante, les équations suivantes doivent être programmées :



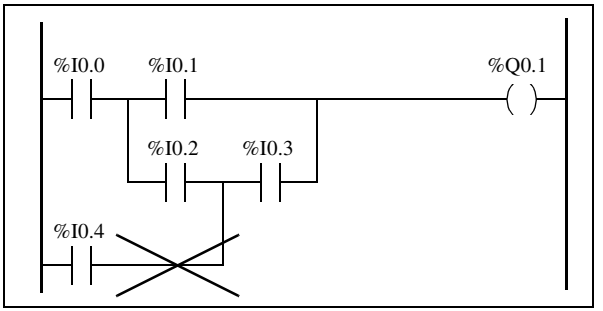

```
LD      %I0.0
MPS
AND(    %I0.1
OR(     %I0.2
AND     %I0.3
)
)
ST      %Q0.1
MPP
AND     %I0.2
ST      %Q0.0
```

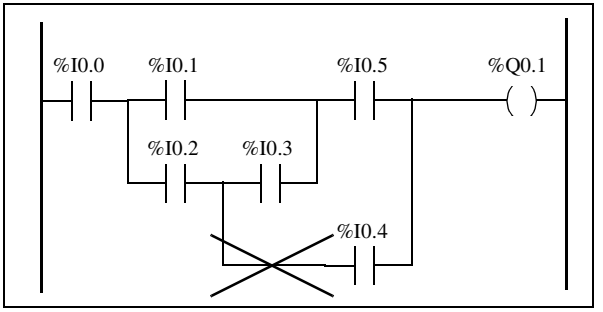
Les contacts placés en parallèle doivent être imbriqués au sein d'un autre contact ou être totalement indépendants les uns des autres :



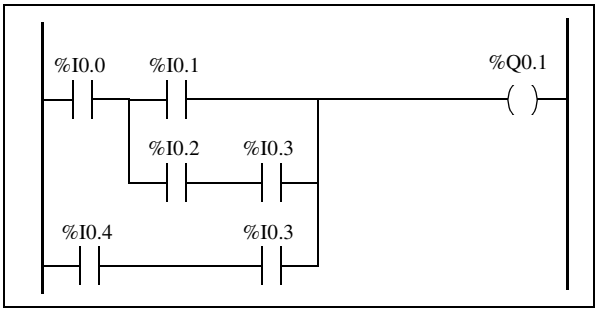


Les schémas suivants ne peuvent pas être programmés :

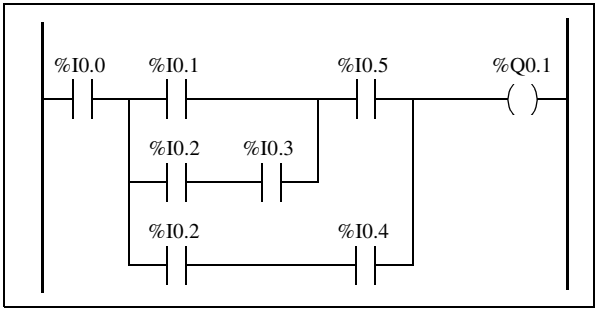




Afin d'exécuter les schémas équivalents, modifiez-les comme illustré ci-dessous :




```
LD      %I0.0
AND(    %I0.1
OR(     %I0.2
AND     %I0.3
)
)
OR(     %I0.4
AND     %I0.3
)
ST      %Q0.1
```




```
LD      %I0.0
AND(    %I0.1
OR(     %I0.2
AND     %I0.3
)
AND     %I0.5
OR(     %I0.2
AND     %I0.4
)
)
ST      %Q0.1
```

Réversibilité schéma à contacts/liste

Introduction

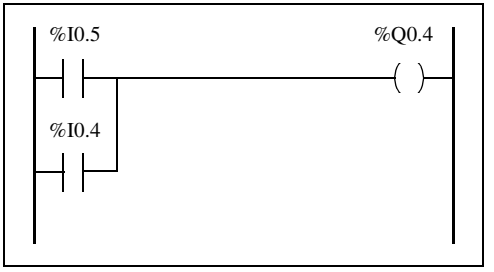
La fonctionnalité de réversibilité du logiciel de programmation TwidoSoft permet de convertir des programmes par schémas à contacts en programmes par listes d'instructions, et vice versa.
Les préférences utilisateur réglées dans TwidoSoft permettent de choisir la méthode d'affichage par défaut des programmes : soit au format liste, soit au format schéma à contacts. TwidoSoft permet également de basculer entre les affichages par liste et par schéma à contacts (reportez-vous au guide d'exploitation TwidoSoft pour plus de détails).

Qu'est-ce que la "réversibilité" ?

Pour bien comprendre à quoi correspond la fonction de réversibilité du programme, il convient d'examiner avec attention les relations existant entre le réseau d'un schéma à contacts et la séquence de la liste d'instructions correspondante :

- **Réseau de schéma à contacts** : ensemble d'instructions par schémas à contacts formant une expression logique.
- **Séquence de liste** : ensemble d'instructions d'un programme par listes, correspondant aux instructions par schémas à contacts et relatif à la même expression logique.

L'illustration suivante présente un réseau de schéma à contacts courant, ainsi que la logique du programme équivalente, exprimée sous la forme d'une liste d'instructions.




```
LD    %I0.5
OR    %I0.4
ST    %Q0.4
```

Un programme d'application est stocké en interne sous la forme d'une liste d'instructions, et ce, que le programme ait été rédigé en langage par schémas à contacts ou par listes. TwidoSoft utilise les similarités de structure de programme existant entre les deux langages, ainsi que l'image liste interne du programme pour l'afficher dans des afficheurs et des éditeurs de listes ou de schémas à contacts, soit sous la forme d'une liste d'instructions (forme élémentaire), soit de manière graphique, sous la forme d'un schéma à contacts, en fonction des préférences sélectionnées par l'utilisateur.

Garantie de réversibilité

Tout programme créé sous forme de schéma à contacts peut être converti en une liste d'instructions. En revanche, certaines logiques du langage par listes ne peuvent pas être converties en langage par schémas à contacts. Pour garantir une réversibilité totale entre le langage par listes et le langage par schémas à contacts, il est important d'observer les directives présentées à la section *Recommandations pour la réversibilité entre le langage schéma à contacts et le langage liste d'instructions, p. 171*.

Recommandations pour la réversibilité entre le langage schéma à contacts et le langage liste d'instructions

Instructions requises pour la réversibilité

La structure d'un bloc fonction réversible dans le langage liste d'instructions requiert l'utilisation des instructions suivantes :

- **BLK** marque le début du bloc et définit le début du réseau, ainsi que celui de la portion d'entrée dans le bloc.
- **OUT_BLK** marque le début de la portion de sortie du bloc.
- **END_BLK** marque la fin du bloc et du réseau.

Il n'est pas nécessaire d'utiliser des instructions de blocs fonctions réversibles pour un programme liste d'instructions qui fonctionne correctement. Certaines instructions permettent une programmation liste d'instructions non réversible. Pour obtenir des informations complètes sur la programmation liste d'instructions non réversible de blocs fonctions, reportez-vous à la rubrique *Principes de programmation de blocs fonctions élémentaires, p. 224*.

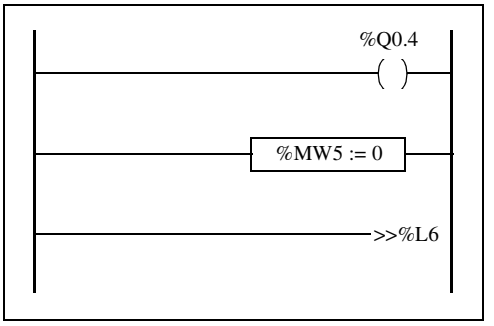
Instructions sans équivalences à éviter

Evitez d'utiliser certaines instructions en langage liste ou certaines associations d'instructions et d'opérandes, pour lesquelles les schémas à contacts ne possèdent pas d'équivalents. Par exemple, l'instruction N (permettant d'inverser la valeur de l'accumulateur booléen) n'a pas d'équivalent dans le langage schémas à contacts. Le tableau suivant répertorie toutes les instructions de programmation liste d'instructions qui ne s'inversent pas dans le langage schéma à contacts :

Instruction par liste	Opérande	Description
JMPCN	%Li	Not saut conditionnel
N	aucun	Négation (Not)
ENDCN	aucun	Not fin conditionnelle

Réseaux inconditionnels

La programmation des réseaux inconditionnels requiert également l'application des recommandations de programmation liste d'instructions suivantes pour que la réversibilité liste d'instructions/schéma à contacts puisse s'opérer. Les réseaux inconditionnels ne sont soumis à aucun test ou à condition. Les sorties ou les instructions d'action sont toujours activées ou exécutées.
Le diagramme suivant présente des exemples de réseaux inconditionnels, ainsi que la séquence en langage liste d'instructions équivalente.




```
LD    1
ST    %Q0.4
LD    1
[%MW5 := 0]
JMP   %L6
```

Vous noterez que chacune des séquences liste d'instructions inconditionnelles ci-dessus commence par une instruction de chargement suivie d'un 1, excepté pour l'instruction JMP. Cette combinaison règle la valeur de l'accumulateur booléen sur 1, et règle par conséquent la bobine (instruction de stockage) sur 1 et %MW5 sur 0 lors de chaque scrutation du programme. L'exception est l'instruction de saut liste inconditionnel (JMP %L6), qui est exécutée quelle que soit la valeur de l'accumulateur et ne nécessite pas le réglage de l'accumulateur sur un.

Réseau schéma à contacts / liste d'instructions

Si un programme liste d'instructions qui n'est pas totalement réversible est inversé, les parties réversibles sont affichées dans la visualisation par schémas à contacts et celles qui sont irréversibles sont affichées sur les réseaux schéma à contacts / liste d'instructions.
Un réseau schéma à contacts / liste d'instructions fonctionne exactement comme un petit éditeur liste d'instructions. Il permet en effet à l'utilisateur de visualiser et de modifier les parties irréversibles d'un programme schéma à contacts.

Documentation du programme

Documentation de votre programme

Vous pouvez documenter votre programme en y ajoutant des commentaires à l'aide des éditeurs de listes et de schémas à contacts (pour plus d'informations sur l'utilisation de ces éditeurs de programmes, reportez-vous au guide d'exploitation TwidoSoft) :

- Dans l'éditeur de listes, des commentaires de lignes vous permettent de documenter votre programme. Ces commentaires peuvent figurer sur la même ligne que les instructions de programmation, ou sur des lignes individuelles distinctes.
- Dans l'éditeur de schémas à contacts, des en-têtes réseau vous permettent de documenter votre programme. Ces en-têtes se situent juste au-dessus du réseau.

Le logiciel de programmation TwidoSoft utilise ces commentaires à des fins de réversibilité. Lors de la conversion d'un programme par listes en programme par schémas à contacts, TwidoSoft utilise certains des commentaires liste pour créer un en-tête réseau. Pour ce faire, les commentaires insérés entre les séquences de liste sont utilisés comme en-têtes réseau.

Exemple de commentaires de ligne de liste

L'exemple suivant illustre un programme par listes possédant des commentaires de lignes.

```
---- ( * TITRE DE L'EN-TETE DU RESEAU 0 * )
---- ( * PREMIER COMMENTAIRE DE L'EN-TETE DU RESEAU 0 * )
---- ( * DEUXIEME COMMENTAIRE DE L'EN-TETE DU RESEAU 0 * )
     0   LD  % I0. 0  ( * COMMENTAIRE DE LIGNE *)
     1   OR  %I0. 1  ( * LIGNE DE COMMENTAIRE IGNOREE LORS DE LA CONVERSION EN SCHEMA À
         CONTACTS * )
     2   ANDM  %M10
     3   ST        M101
---- ( * EN-TETE DU RESEAU 1 * )
---- ( * CE RESEAU CONTIENT UNE ETIQUETTE * )
---- ( * DEUXIEME COMMENTAIRE DE L'EN-TETE DU RESEAU 1 * )
---- ( * TROISIEME COMMENTAIRE DE L'EN-TETE DU RESEAU 1 * )
---- ( * QUATRIEME COMMENTAIRE DE L'EN-TETE DU RESEAU 1 * )
     4   % L5:
     5   LD  %M101
     6   [ %MW20 := %KW2 * 16]
---- ( * CE RESEAU NE CONTIENT QUE LE TITRE D'UN EN-TETE * )
     7   LD  %Q0. 5
     8   OR  %I0.  3
     9   ORR  I0.  13
    10   ST  %Q0.5
```

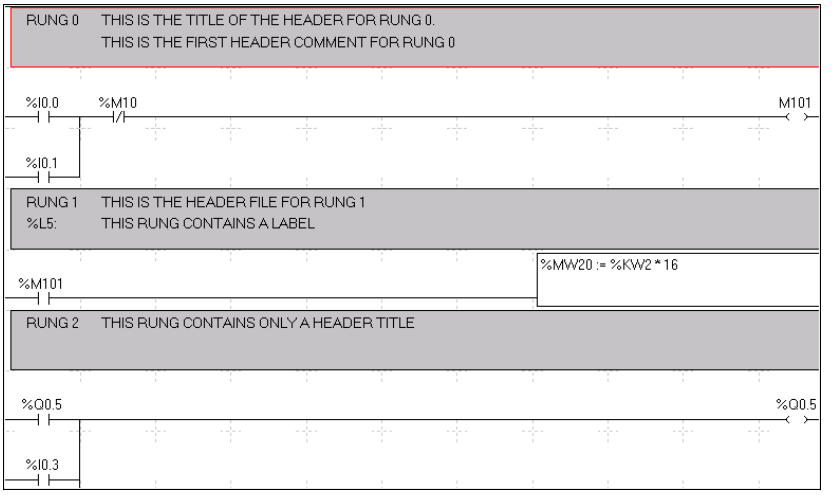
Conversion de commentaires de liste en en-tête réseau de schéma à contacts

Lorsque qu'un programme par listes est converti en programme par schémas à contacts, les commentaires de ligne de liste sont affichés dans l'éditeur de schémas à contacts en fonction des règles suivantes :

- Le premier commentaire figurant sur une ligne individuelle est utilisé comme en-tête réseau.
- Les commentaires suivants sont utilisés pour former le corps du réseau.
- Lorsque les lignes du corps de l'en-tête sont toutes remplies, les commentaires de ligne compris entre les séquences de liste sont ignorés, tout comme les autres commentaires situés dans des lignes de liste et qui contiennent également des instructions.

Exemple de commentaires d'en-têtes réseau

L'exemple suivant illustre un programme par schémas à contacts possédant des commentaires d'en-têtes réseau.



Conversion de commentaires de schémas à contacts en commentaires de listes

Lorsqu'un schéma à contacts est converti en une liste d'instructions, les commentaires d'en-têtes réseau sont affichés dans l'éditeur de listes en fonction des règles suivantes :

- Tous les commentaires d'en-tête réseau sont insérés entre les séquences de liste associées.
- Toutes les étiquettes (%Li:) et les déclarations de sous-programme (SRi:) sont placées sur la ligne suivant l'en-tête et précédant immédiatement la séquence de liste.
- Si le programme avait déjà été converti du format liste au format schéma à contacts, tous les commentaires précédemment ignorés seront de nouveau affichés dans l'éditeur de listes.

Langage liste d’instructions

10

En bref...

Présentation

Cette rubrique décrit la programmation à l'aide du langage liste d'instructions.

Contenu de ce chapitre

Ce chapitre contient les sujets suivants :

Vue d'ensemble des programmes en langage liste d'instructions

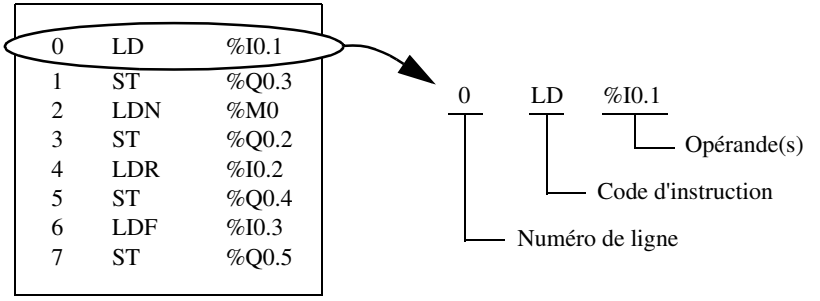
Introduction

Un programme écrit en langage liste d'instructions est constitué d'une série d'instructions exécutées en séquence par l'automate. Chaque instruction est représentée par une seule ligne de code et se compose de trois éléments :

- Numéro de ligne
- Code d'instruction
- Opérande(s)

Exemple de programme liste d'instructions

L'illustration suivante est un exemple de programme liste d'instructions.



Numéro de ligne

Les numéros de ligne sont générés automatiquement lorsque vous saisissez une instruction. Les lignes vides et les lignes de commentaires n'ont pas de numéro de ligne.

Code d'instruction

Le code d'instruction est un symbole désignant un opérateur qui identifie l'opération à effectuer à l'aide des opérandes. Les opérateurs types spécifient les opérations booléennes et numériques.
Par exemple, dans l'échantillon de programme présenté ci-dessus, LD est l'abréviation de LOAD en code d'instruction. L'instruction LOAD place (charge) la valeur de l'opérande %I0.1 dans un registre interne nommé accumulateur.
Il existe deux types d'instructions de base :

- Instructions de test
  Il s'agit de réglages ou de tests des conditions nécessaires à l'accomplissement d'une action. Par exemple, LOAD (LD) et AND.
- Instructions d'action
  Elles permettent d'effectuer les actions autorisées lorsque les conditions de configuration sont remplies. Par exemple, des instructions d'affectation telles que STORE (ST) et RESET (R).

Opérande

Un opérande est un nombre, un repère ou un symbole représentant une valeur qu'un programme peut manipuler au sein d'une instruction. Par exemple, dans l'échantillon de programme présenté ci-dessus, l'opérande %I0.1 est un repère auquel on a affecté la valeur d'une entrée de l'automate. Une instruction peut avoir entre zéro et trois opérandes selon le type de code d'instruction.
Les opérandes peuvent représenter les éléments suivants :

- les entrées/sorties de l'automate, telles que les capteurs, boutons poussoirs et relais ;
- les fonctions système prédéfinies, telles que les temporisateurs et les compteurs ;
- les opérations arithmétiques, logiques, de comparaisons et numériques ;
- les variables internes de l'automate, telles que les bits et les mots.

Fonctionnement des listes d'instructions

Introduction

Les listes d'instructions ne possèdent qu'une seule opérande explicite, l'autre étant implicite. L'opérande implicite correspond à la valeur de l'accumulateur booléen. Par exemple, dans l'instruction LD %I0.1, %I0.1 est l'opérande explicite. Une opérande implicite est stockée dans l'accumulateur et se voit écrasée par la valeur de %I0.1.

Fonctionnement

Une instruction en langage liste d'instructions exécute une opération spécifiée sur le contenu de l'accumulateur et sur l'opérande explicite, puis remplace le contenu de l'accumulateur par le résultat obtenu. Par exemple, l'opération AND %I1.2 effectue un AND logique entre le contenu de l'accumulateur et celui de l'entrée 1.2 et remplace le contenu de l'accumulateur par ce résultat.
L'ensemble des instructions booléennes, à l'exception des instructions de chargement, de stockage et les instructions NOT, fonctionnent avec deux opérandes. La valeur des deux opérandes peut être True ou False et l'exécution des instructions par le programme génère une valeur unique : soit True, soit False. Les instructions de chargement placent la valeur de l'opérande dans l'accumulateur, tandis que les instructions de stockage transfèrent la valeur de l'accumulateur vers l'opérande. L'instruction NOT ne possède aucune opérande explicite et a seulement pour effet d'inverser l'état de l'accumulateur.

Instructions en langage liste d'instructions prises en charge

Le tableau suivant répertorie les différents types d'instructions en langage liste d'instructions prises en charge :

Type d'instruction	Exemple	Fonction
Instruction Bit	LD %M10	Lit le bit interne %M10
Intruction sur bloc	IN %TM0	Démarre le temporisateur %TM0
Instruction sur mot	[%MW10 := %MW50+100]	Opération d'addition
Instruction sur programme	SR5	Appelle le sous-programme n°5
Instruction Grafcet	-*-8	Etape n°8

Instructions en langage liste d'instructions

Introduction

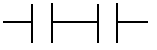
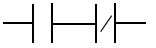
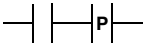
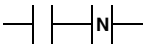
Le langage liste d'instructions comprend les types d'instructions suivants :

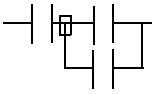
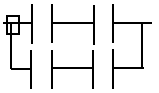
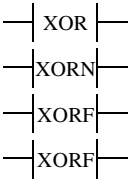
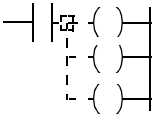
- Instructions de test
- Instructions d'action
- Instructions de blocs fonctions

Cette rubrique identifie et décrit les instructions Twido de programmation en langage liste d'instructions.

Instructions de test

Le tableau suivant décrit les instructions de test du langage liste d'instructions.

Nom	Elément graphique correspondant	Fonction
LD		Le résultat booléen correspond à l'état de l'opérande.
LDN		Le résultat booléen correspond à l'état inversé de l'opérande.
LDR		Le résultat booléen prend la valeur 1 lorsque le passage de l'opérande (front montant) de 0 à 1 est détecté.
LDF		Le résultat booléen devient 1 lorsque le passage de l'opérande (front descendant) de 1 à 0 est détecté.
AND		Le résultat booléen est égal à la logique AND entre le résultat booléen de l'instruction précédente et l'état de l'opérande.
ANDN		Le résultat booléen est égal à la logique AND entre le résultat booléen de l'instruction précédente et l'état inversé de l'opérande.
ANDR		Le résultat booléen est égal à la logique AND entre le résultat booléen de l'instruction précédente et la détection du front montant de l'opérande (1 = front montant).
ANDF		Le résultat booléen est égal à la logique AND entre le résultat booléen de l'instruction précédente et la détection du front descendant de l'opérande (1 = front descendant).
OR		Le résultat booléen est égal à la logique OR entre le résultat booléen de l'instruction précédente et l'état de l'opérande.

Nom	Elément graphique correspondant	Fonction
AND(		Logique AND (8 niveaux de parenthèses)
OR(		Logique OR (8 niveaux de parenthèses)
XOR, XORN, XORR, XORF		OR exclusif
MPS MRD MPP		Commutation vers les bobines
N	-	Négation (NOT)

Instructions d'action

Le tableau suivant décrit les instructions d'action du langage liste d'instructions.

Nom	Elément graphique correspondant	Fonction
ST	—( )—	L'opérande associée prend la valeur du résultat de la zone de test.
STN	—(/)—	L'opérande associée prend la valeur inversée du résultat de la zone de test.
S	—(S)—	L'opérande associée est réglée sur 1 lorsque le résultat de la zone de test est 1.
R	—(R)—	L'opérande associée est réglée sur 0 lorsque le résultat de la zone de test est 1.
JMP	->>%Li	Se connecte inconditionnellement à une séquence portant une étiquette, en amont ou en aval.
SRn	->>%SRi	Connexion au début d'un sous-programme.
RET	<RET>	Retour d'un sous-programme.
END	<END>	Fin de programme.
ENDC	<ENDC>	Fin du programme conditionné avec un résultat booléen de 1.
ENDCN	<ENDCN>	Fin du programme conditionné avec un résultat booléen de 0.

Instructions de blocs fonctions

Le tableau suivant décrit les instructions de blocs fonctions du langage liste d'instructions.

Nom	Elément graphique correspondant	Fonction
Temporisateurs, compteurs, registres, etc.		Il existe des instructions de régulation de bloc pour chaque bloc fonction. Une forme structurée est utilisée pour raccorder directement les entrées et les sorties du bloc. **Remarque :** Les sorties des blocs fonctions ne peuvent pas être connectées les unes aux autres (liaisons verticales).

Utilisation de parenthèses

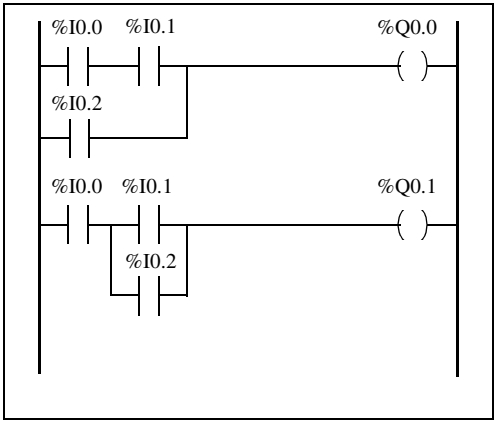
Introduction

Dans les instructions logiques AND et OR, les parenthèses permettent de spécifier des divergences dans des schémas à contacts. Les parenthèses ouvrante et fermante sont associées à des instructions, de la manière suivante :

- L'ouverture des parenthèses est associée à l'instruction AND ou OR.
- La fermeture des parenthèses correspond à une instruction requise pour chaque parenthèse ouvrante.

Exemple d'utilisation d'une instruction AND

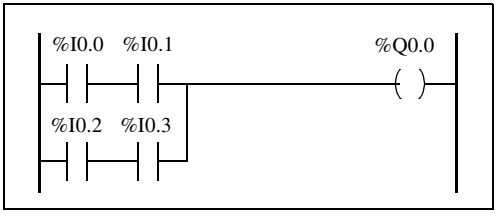
Les schémas suivants illustrent l'utilisation des parenthèses dans une instruction AND : AND(...).




```
LD      %I0.0
AND     %I0.1
OR      %I0.2
ST      %Q0.0

LD      %I0.0
AND(    %I0.1
OR      %I0.2
)
ST      %Q0.1
```

Exemple d'utilisation d'une instruction OR

Les schémas suivants illustrent l'utilisation des parenthèses dans une instruction OR : OR(...).




```
LD      %I0.0
AND     %I0.1
OR(     %I0.2
AND     %I0.3
)
ST      %Q0.0
```

Modificateurs

Le tableau suivant répertorie les modificateurs pouvant être affectés à des parenthèses.

Modificateur	Fonction	Exemple
N	Négation	AND(N ou OR(N
F	Front descendant	AND(F ou OU(F
R	Front montant	AND(R ou OU(R
[	Comparaison	Reportez-vous à la rubrique *Instructions de comparaison, p. 252*

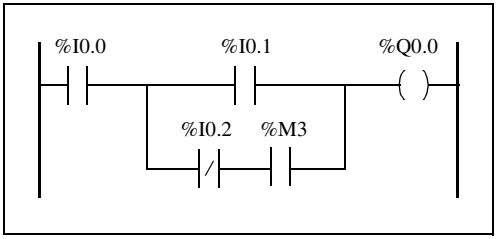
Imbrication de parenthèses

Il est possible d'imbriquer un maximum de huit niveaux de parenthèses.
Veuillez appliquer les règles suivantes lors de l'imbrication de parenthèses :

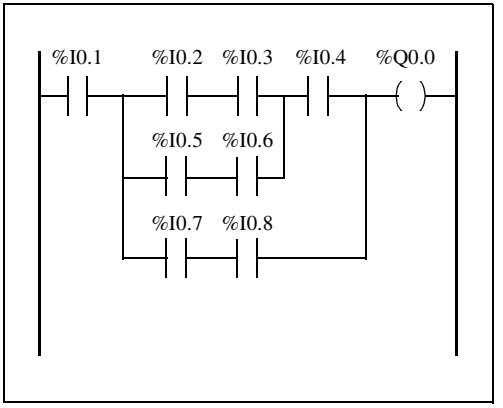
- Une parenthèse fermante doit être insérée pour chaque parenthèse ouvrante.
- Les étiquettes (%Li:), les sous-programmes (SRi:), les instructions de saut (JMP) et les instructions de bloc fonction ne doivent pas être placés dans des expressions comprises entre parenthèses.
- Les instructions de stockage ST, STN, S et R ne doivent pas être programmées entre parenthèses.
- Les instructions de pile MPS, MRD et MPP ne peuvent pas être utilisées entre parenthèses.

Exemples d'imbrication de parenthèses

Les schémas suivants illustrent l'imbrication de parenthèses.




```
LD      %I0.0
AND(    %I0.1
OR(N    %I0.2
AND     %M3
)
)
ST      %Q0.0
```




```
LD      %I0.1
AND(    %I0.2
AND     %I0.3
OR(     %I0.5
AND     %I0.6
)
AND     %I0.4
OR(     %I0.7
AND     %I0.8
)
)
ST      %Q0.0
```

Instructions de pile (MPS, MRD, MPP)

Introduction

Les instructions de pile permettent de traiter le routage vers des bobines .Les instructions MPS, MRD et MPP utilisent une zone de stockage temporaire appelée « pile ». Cette pile peut stocker un maximum de huit expressions booléennes.

Note : Ces instructions ne peuvent pas être utilisées dans une expression comprise entre parenthèses.

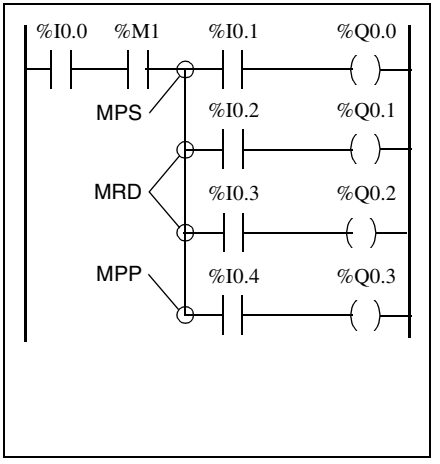
Fonctionnement des instructions de pile

Le tableau suivant décrit le fonctionnement des trois instructions de pile.

Instruction	Description	Fonction
MPS	Abréviation de Memory Push onto Stack (Push mémoire sur la pile)	Stocke le résultat de la dernière instruction logique (contenu de l'accumulateur) en haut de la pile. Ceci a pour effet de décaler les autres valeurs de la pile vers le bas.
MRD	Abréviation de Memory Read from stack (Lecture mémoire depuis la pile)	Lit la valeur stockée en haut de la pile et la transmet à l'accumulateur.
MPP	Abréviation de Memory Pop from Stack (Extraction mémoire depuis la pile)	Coupe la valeur située dans le haut de la pile, la transmet à l'accumulateur et déplace les autres valeurs de la pile vers le haut.

Exemples d'instructions de pile

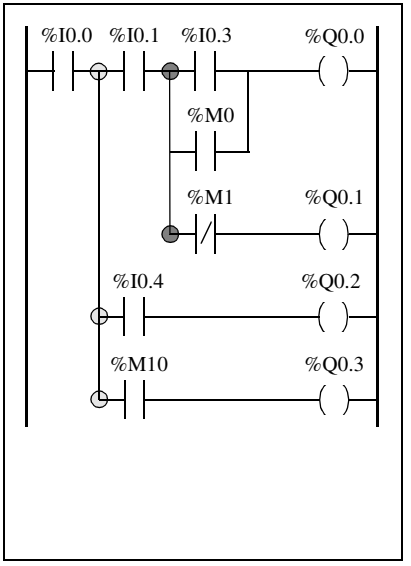
Les schémas suivants illustrent l'utilisation d'instructions de pile.

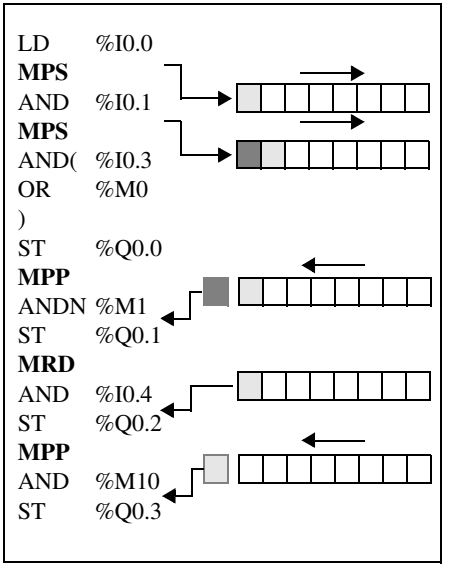



```
LD      %I0.0
AND     %M1
MPS
AND     %I0.1
ST      %Q0.0
MRD
AND     %I0.2
ST      %Q0.1
MRD
AND     %I0.3
ST      %Q0.2
MPP
AND     %I0.4
ST      %Q0.3
```

Exemples du fonctionnement de la pile

Les schémas suivants illustrent le fonctionnement des instructions de pile.





Grafcet

11

En bref...

Présentation

Cette rubrique décrit la programmation à l'aide du langage Grafcet.

Contenu de ce chapitre

Ce chapitre contient les sujets suivants :

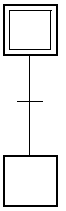
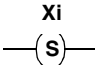
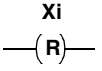
Description des instructions Grafcet

Introduction

Les instructions Grafcet de TwidoSoft offrent une méthode simple de traduction de séquences de régulation (graphe Grafcet).
Le nombre maximum d'étapes Grafcet dépend du type d'automate Twido. Le nombre d'étapes pouvant être activées simultanément est uniquement limité par le nombre total d'étapes.
Pour les automates TWDLCAA10DRF et TWDLCAA16DRF, les étapes 1 à 62 sont disponsibles. Pour tous les autres automates, les étapes 1 à 94 sont disponibles.

Instructions Grafcet

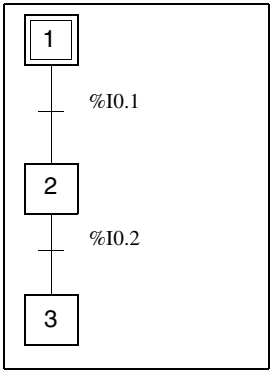
Le tableau suivant répertorie toutes les instructions et les objets requis pour la programmation d'un graphe Grafcet.

Représentation graphique (1)	Transcription dans le langage TwidoSoft	Fonction
	=*= i	Lance l'étape initiale (2).
	# i	Active l'étape i après avoir désactivé l'étape courante.
	-*- i	Lance l'étape i et valide la transition associée (2).
	#	Désactive l'étape courante sans activer d'autre étape.
	#Di	Désactive l'étape i et l'étape courante.
	=*= POST	Lance le traitement postérieur et termine le traitement séquentiel.
	%Xi	Bit associé à l'étape i. Peut être testé et écrit (le nombre maximum d'étapes dépend de l'automate).
	LD %Xi, LDN %Xi AND %Xi, ANDN %Xi, OR %Xi, ORN %Xi XOR %Xi, XORN %Xi	Teste l'activité de l'étape i.
	S %Xi	Active l'étape i.
	R %Xi	Désactive l'étape i.

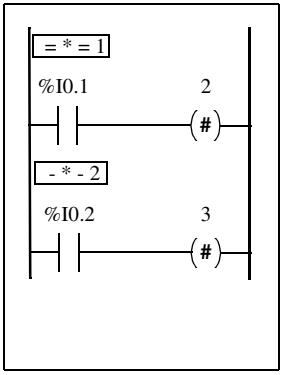
(1) Graphical Grafcet n'est pas pris en charge.
(2) La première étape =*=i ou -*-i écrite indique le lancement du traitement séquentiel et, par conséquent, la fin du pré-traitement.

Exemples Grafcet

Séquence linéaire :



Non pris en charge

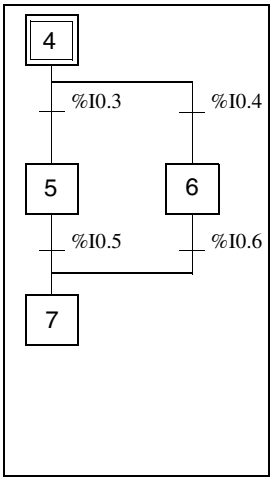


Programme schéma à contacts Twido

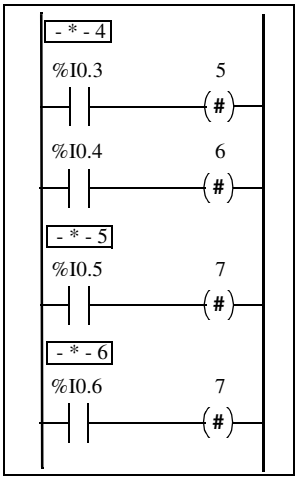
```
=*=     1
LD      %I0.1
 #      2

-*-     2
LD      %I0.2
 #      3
```

Programme liste d'instructions Twido

Séquence de divergences :



Non pris en charge

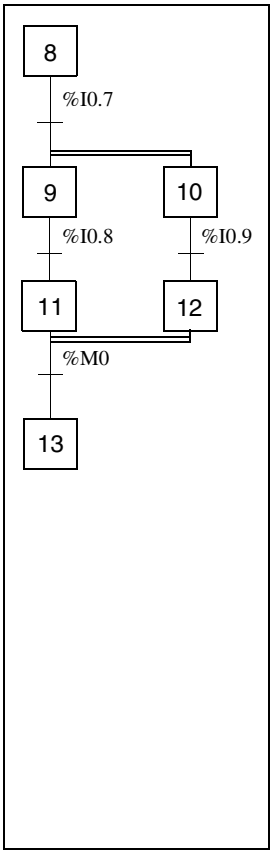


Programme schéma à contacts Twido

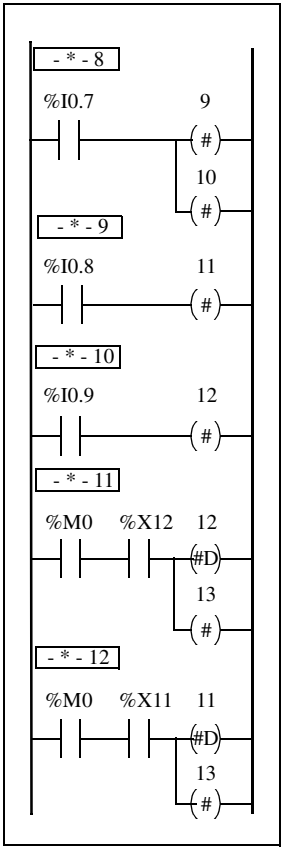
```
-*-     4
 LD     %I0.3
 #      5
 LD     %I0.4
 #      6

-*-     5
 LD     %I0.5
 #      7

-*-     6
 LD     %I0.6
 #      7
```

Programme liste d'instructions Twido

Séquences simultanées :



Non pris en charge



Programme schéma à contacts Twido

```
 -*-      8
 LD       %I0.7
 #        9
 #        10

-*-       9
 LD       %I0.8
 #        11

-*-       10
 LD       %I0.9
 #        12

-*-       11
 LD       %M0
 AND      %X12
 #D       12
 #        13

-*-       12
 LD       %M0
 AND      %X11
 #D       11
 #        13
```

Programme liste d'instructions Twido

Note : Pour qu'un graphe Grafcet soit opérationnel, au moins une étape active doit être déclarée à l'aide de l'instruction =*=i (étape initiale) ou le graphe doit être prépositionné lors du pré-traitement à l'aide du bit système %S23 et de l'instruction S %Xi.

Description de la structure d'un programme Grafcet

Introduction

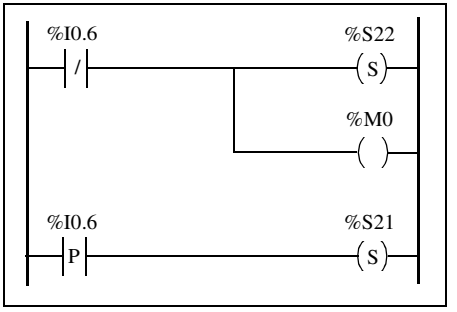
Un programme TwidoSoft Grafcet se déroule en trois phases :

- Pré-traitement
- Traitement séquentiel
- Traitement postérieur

Pré-traitement

Le pré-traitement gère les éléments suivants :
- les reprises de l'alimentation ;
- les défauts ;
- les changements du mode de fonctionnement ;
- le pré-positionnement des étapes Grafcet ;
- la logique d'entrée.

Dans l'exemple de pré-traitement ci-dessous (zone préalable à la première étape Grafcet), l'état 0 de l'entrée %I0.6 vous invite à réinitialiser le graphe Grafcet en réglant le bit système %S22 sur 1. Ceci a pour effet de désactiver les étapes actives. Le front montant de l'entrée %I0.6 prépositionne le graphe sur l'étape X1. Enfin, l'utilisation du bit système %S21 force l'initialisation de Grafcet.




```
000   LDN   %I0.6
001   S     %S22
002   ST    %M0
003   LDR   %I0.6
004   S     %S21
```

Le pré-traitement commence à la première ligne du programme et se termine à la première occurrence d'une instruction "= * =" ou "- * -".
Trois bits système sont dédiés à la régulation Grafcet : %S21, %S22 et %S23. Chaque bit système est réglé sur 1 (si nécessaire) par l'application, lors du pré-traitement généralement. La fonction associée est exécutée par le système à la fin du pré-traitement et le bit système est remis à 0 par le système.

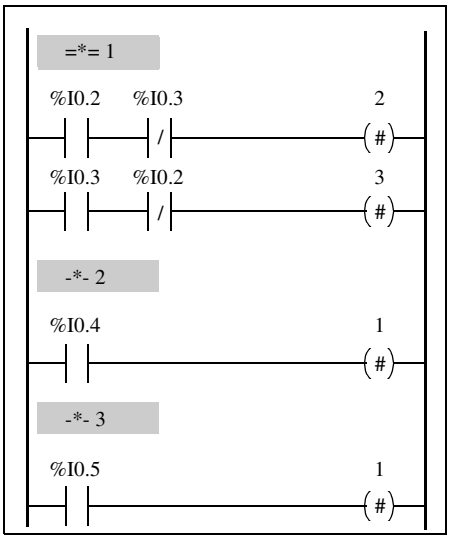
Bit système	Nom	Description
%S21	Initialisation de Grafcet	Toutes les étapes actives sont désactivées et les étapes initiales sont activées.
%S22	Réinitialisation de Grafcet	Toutes les étapes sont désactivées.
%S23	Prépositionnement de Grafcet	Ce bit doit être réglé sur 1 si les objets %Xi sont explicitement écrits par l'application lors du pré-traitement. Si ce bit est maintenu sur 1 lors du pré-traitement sans changement explicite des objets %Xi, Grafcet est figé (aucune mise à jour n'est prise en compte).

Traitement séquentiel

Le traitement séquentiel est exécuté dans le graphe (instructions représentant le graphe) :

- étapes
- actions associées aux étapes
- transitions
- conditions de transition

Exemple :




```
005     =*=     1
006     LD      %I0.2
007     ANDN    %I0.3
008     #       2
009     LD      %I0.3
010     ANDN    %I0.2
011     #       3
012     -*-     2
013     LD      %I0.4
014     #       1
015     -*-     3
016     LD      %I0.5
017     #       1
```

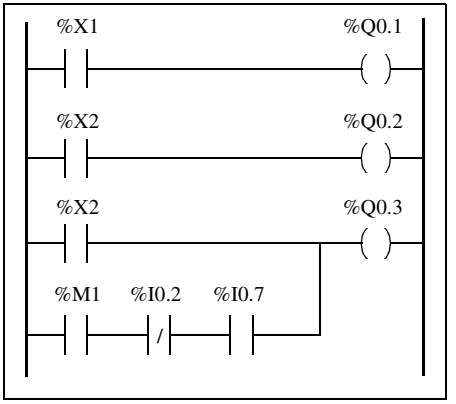
Le traitement séquentiel se termine par l'exécution de l'instruction "= * = POST" ou par la fin du programme.

Traitement postérieur

Le traitement postérieur gère les éléments suivants :

- les commandes du traitement séquentiel pour la régulation des sorties ;
- le verrouillage de sécurité spécifique aux sorties.

Exemple :




```
018   =*=    POST
019   LD     %X1
020   ST     %Q0.1
021   LD     %X2
022   ST     %Q0.2
023   LD     %X3
024   OR(    %M1
025   ANDN   %I0.2
026   AND    %I0.7
027   )
028   ST     %Q0.3
```

Actions associées aux étapes Grafcet

Introduction

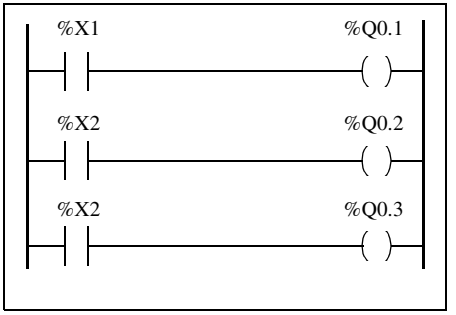
Un programme Grafcet TwidoSoft offre deux modes de programmation des actions associées aux étapes :

- dans la section de traitement postérieur ;
- dans les listes d'instructions ou les réseaux schéma à contacts des étapes mêmes.

Association des actions dans le traitement postérieur

Si des contraintes de sécurité ou de mode d'exécution sont appliquées, il est préférable de programmer les actions dans la section de traitement postérieur d'une application Grafcet. Vous pouvez utiliser les instructions en langage liste d'instructions SET et RESET ou activer les bobines d'un programme schéma à contacts pour lancer les étapes Grafcet (%Xi).

Exemple :

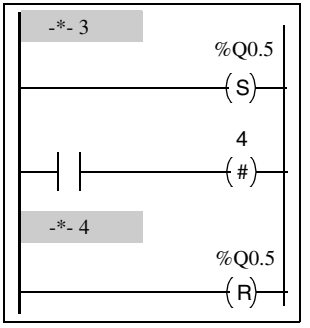



```
018     =*=     POST
019     LD      %X1
020     ST      %Q0.1
021     LD      %X2
022     ST      %Q0.2
023     LD      %X3
024     ST      %Q0.3
```

Association d'actions à partir d'une application

Vous pouvez programmer les actions associées aux étapes sous forme de listes d'instructions ou de réseaux schéma à contacts. Dans ce cas, la liste d'instructions ou le réseau schéma à contacts n'est pas scruté(e), à moins que l'étape ne soit active. Ce mode d'utilisation du langage Grafcet est le plus efficace, le plus lisible et le plus facile à gérer.

Exemple :




```
020     -*-     3
021     LD      1
022     S       %Q0.5
023     LD      %M10
024     #       4
025     -*-     4
026     LD      1
027     R       %Q0.5
028     ...
029     ...
```

Description des instructions et des fonctions

IV

En bref...

Présentation

Cette rubrique fournit des descriptions détaillées des instructions élémentaires et avancées, ainsi que des bits et des mots système des langages Twido.

Contenu de cet intercalaire

Cet intercalaire contient les chapitres suivants :

Instructions élémentaires

12

En bref...

Présentation

Cette rubrique fournit des détails sur les instructions et les blocs fonctions utilisés pour créer des programmes de régulation élémentaires des automates Twido.

Contenu de ce chapitre

Ce chapitre contient les sous-chapitres suivants :

12.1 Traitement booléen

Introduction au traitement booléen

Présentation

Cette rubrique offre une introduction au traitement booléen. Elle s'appuie sur des descriptions et des directives de programmation d'instructions booléennes.

Contenu de ce sous-chapitre

Ce sous-chapitre contient les sujets suivants :

Instructions booléennes

Introduction

Les instructions booléennes s'apparentent aux éléments graphiques du langage schéma à contacts. Ces instructions sont présentées dans le tableau suivant.

Elément	Instruction	Exemple	Description
Eléments de test	L'instruction de chargement (LD) équivaut à un contact ouvert.	LD %I0.0	Le contact est fermé lorsque le bit de régulation se trouve à l'état 1.
Eléments d'action	L'instruction de stockage (ST) équivaut à une bobine.	ST %Q0.0	L'objet bit associé prend la valeur logique de l'accumulateur de bit (résultat de la logique précédente).

Le résultat booléen des éléments de test est appliqué aux éléments d'action, comme l'illustrent les instructions suivantes.

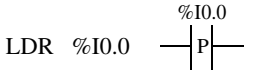
```
LD    %I0.0
AND   %I0.1
ST    %Q0.0
```

Test des entrées de l'automate

Des instructions de test booléennes peuvent être utilisées pour détecter des fronts montants ou descendants sur les entrées de l'automate. Un front est détecté lorsque l'état d'une entrée est passé de la valeur « scan n-1 » à la valeur « scan n » courante. La détection de ce front reste effective pendant la scrutation courante.

Détection d'un front montant

L'instruction LDR (Load Rising Edge - chargement du front montant) équivaut à un contact de détection d'un front montant. Le front montant détecte le passage des valeurs des entrées de régulation de 0 à 1.
Un contact de détection de transition positive est utilisé pour détecter un front montant, comme l'illustre le schéma suivant.



P : contact de détection de transition positive

Détection d'un front descendant

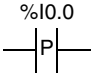
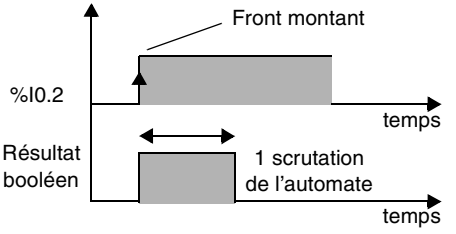
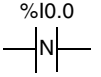
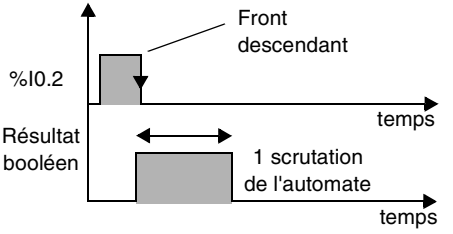
L'instruction LDF (Load Falling Edge - chargement du front descendant) équivaut à un contact de détection d'un front descendant. Le front descendant détecte le passage de la valeur de l'entrée de régulation de 1 à 0.
Un contact de détection de transition négative est utilisé pour détecter un front descendant, comme l'illustre le schéma suivant.

LDF %I0.0 

N : contact de détection de transition négative

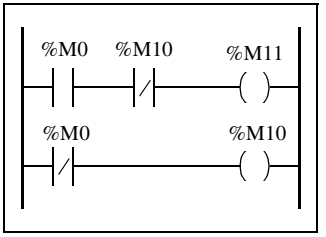
Temporisation de détection d'un front

Le tableau suivant résume les instructions, ainsi que la temporisation des instructions booléennes utilisées pour détecter les fronts montants et descendants.

Front	Instruction de test	Schéma à contacts	Temporisation
Front montant	LDR %I0.0		
Front descendant	LDF %I0.0		

Utilisation de bits internes pour la détection d'un front

Les instructions placées sur un front montant ou descendant s'appliquent aux entrées %I. Il est cependant possible de détecter des fronts sur tout autre bit (ou sur tout autre résultat booléen), à l'aide de deux bits internes.
Dans l'exemple suivant, le bit %M11 enregistre le front montant du bit %M0.




```
LD      %M0
ANDN    %M10
ST      %M11
LDN     %M0
ST      %M10
```

Note : Lors d'un démarrage à froid ou d'une reprise à chaud, l'application détecte un front montant même si la valeur de l'entrée est restée sur 1. Ceci peut être masqué en démarrant le programme sur des instructions LD %S1 et ENDC.

Note : Seuls les bits d'entrée (%Ii) permettent de détecter directement des fronts montants ou descendants.

Explication du format de description des instructions booléennes

Introduction

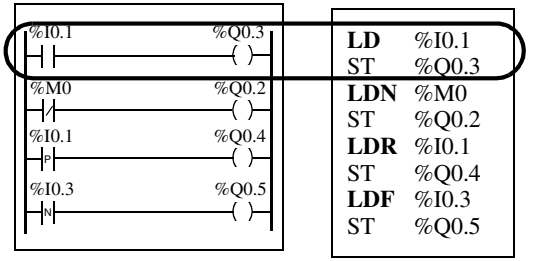
Chaque instruction booléenne de cette rubrique est décrite à l'aide des informations suivantes :

- Description rapide
- Exemple représentant l'instruction et schéma à contacts correspondant
- Liste d'opérandes autorisées
- Chronogramme

Les explications ci-dessous présentent plus en détails le mode de description des instructions booléennes dans cette rubrique.

Exemples

L'illustration suivante présente le mode d'affichage des exemples pour chaque instruction.



Équivalents dans le langage schéma à contacts

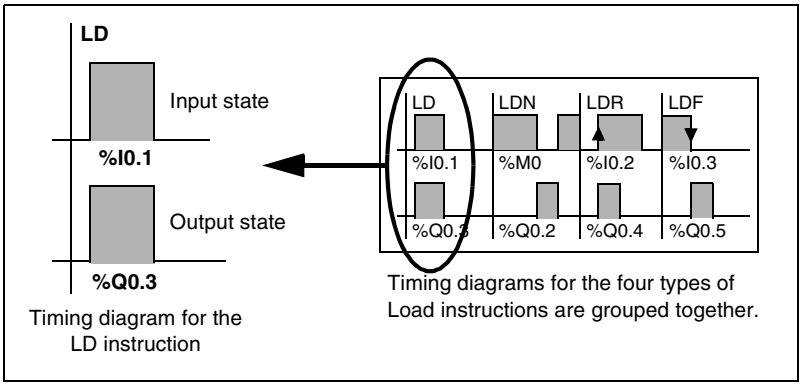
Instructions en langage liste d'instructions

Opérandes autorisées

Le tableau suivant définit les types d'opérandes autorisées utilisées dans les instructions booléennes.

Opérande	Description
0/1	Valeur immédiate de 0 ou 1
%I	Entrée automate %Ii.j
%Q	Sortie automate %Qi.j
%M	Bit interne %Mi
%S	Bit système %Si
%X	Bit d'étape %Xi
%BLK.x	Bit de bloc fonction (%TMi.Q, par exemple)
%•:Xk	Bit de mot (%MWi:Xk, par exemple)
[	Expression de comparaison ([%MWi<1000], par exemple)

Chronogrammes L'illustration suivante présente le mode d'affichage des chronogrammes pour chaque instruction.



Timing diagram for the LD instruction

Timing diagrams for the four types of Load instructions are grouped together.

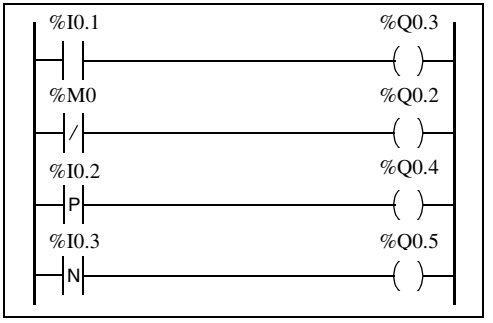
Instructions de chargement (LD, LDN, LDR, LDF)

Introduction

Les instructions de chargement LD, LDN, LDR et LDF correspondent respectivement aux contacts Ouvert, Fermé, Front montant et Front descendant (les instructions LDR et LDF ne sont utilisées qu'avec des entrées de l'automate).

Exemples

Les schémas suivants sont des exemples d'instructions de chargement.



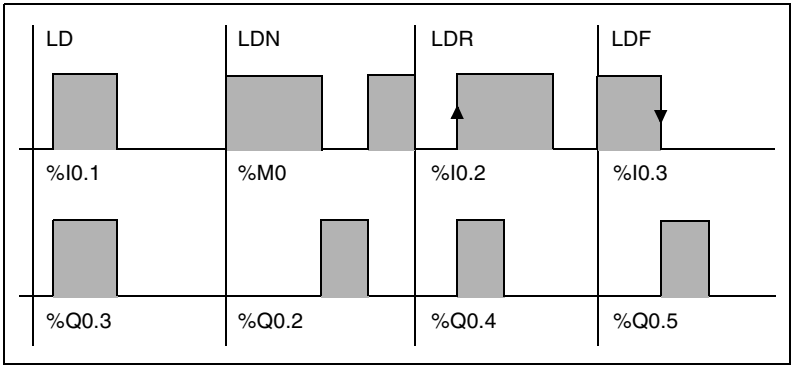

```
LD    %I0.1
ST    %Q0.3
LDN   %M0
ST    %Q0.2
LDR   %I0.2
ST    %Q0.4
LDF   %I0.3
ST    %Q0.5
```

Opérandes autorisés

Le tableau suivant répertorie les types d'instructions de chargement, leurs équivalents dans le langage schéma à contacts, ainsi que les opérandes autorisés.

Instruction par liste	Symbole équivalent dans un schéma à contacts	Opérandes autorisés
LD		0/1,%I,%Q,%M,%S,%X,%BLK.x,%•:Xk,[
LDN		%I,%Q,%M,%S,%X,%BLK.x,%•:Xk,[
LDR		%I
LDF		%I

Chronogramme Le diagramme suivant illustre la temporisation des instructions de chargement.



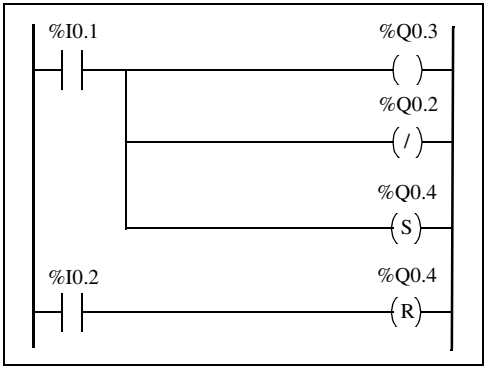
Instructions de stockage (ST, STN, R, S)

Introduction

Les instructions de stockage ST, STN, S et R correspondent respectivement aux bobines Directe, Inverse, SET et RESET.

Exemples

Les schémas suivants sont des exemples d'instructions de stockage.



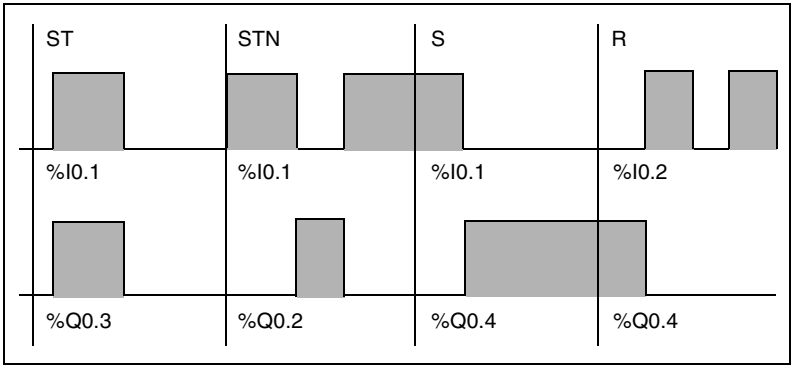

```
LD      %I0.1
ST      %Q0.3

STN     %Q0.2
S       %Q0.4

LD      %I0.2
R       %Q0.4
```

Opérandes autorisés

Le tableau suivant répertorie les types d'instructions de stockage, leurs équivalents dans le langage schéma à contacts, ainsi que les opérandes autorisés.

Instruction par liste	Symbole équivalent dans un schéma à contacts	Opérandes autorisés
ST	( )	%Q,%M,%S,%BLK.x,%•:Xk
STN	(/)	%Q,%M,%S,%BLK.x,%•:Xk
S	(S)	%Q,%M,%S,%X,%BLK.x,%•:Xk
R	(R)	%Q,%M,%S,%X,%BLK.x,%•:Xk

Chronogramme Le diagramme suivant illustre la temporisation des instructions de stockage.



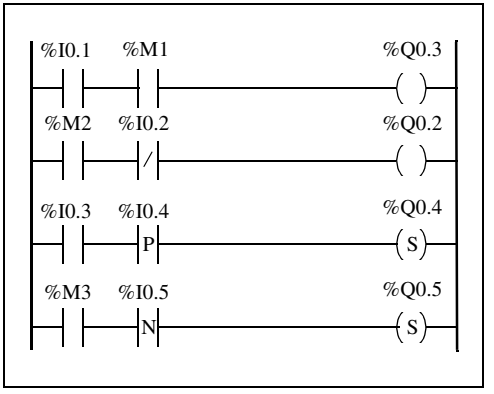
Instructions AND logique (AND, ANDN, ANDR, ANDF)

Introduction

Les instructions AND effectuent une opération de liaison AND logique entre l'opérande (ou son inverse, ou son front montant ou descendant) et le résultat booléen de l'instruction précédente.

Exemples

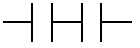
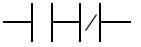
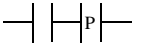
Les schémas suivants sont des exemples d'instructions AND.



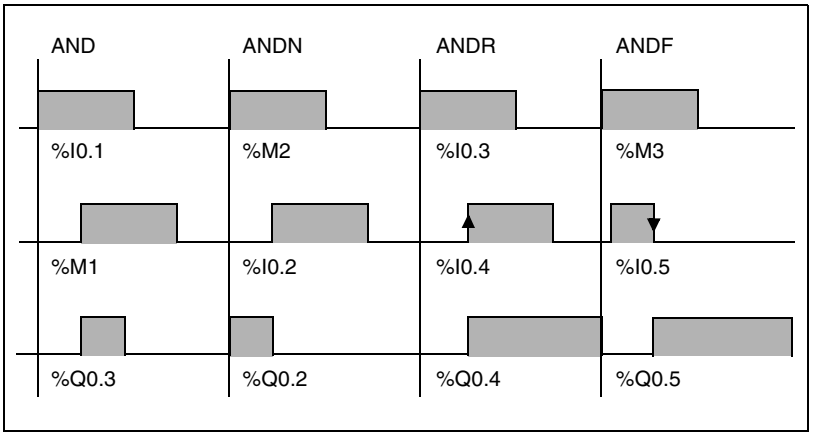

```
LD      %I0.1
AND     %M1
ST      %Q0.3
LD      %M2
ANDN    %I0.2
ST      %Q0.2
LD      %I0.3
ANDR    %I0.4
S       %Q0.4
LD      %M3
ANDF    %I0.5
S       %Q0.5
```

Opérandes autorisés

Le tableau suivant répertorie les types d'instructions AND, leurs équivalents dans le langage schéma à contacts, ainsi que les opérandes autorisés.

Instruction par liste	Symbole équivalent dans un schéma à contacts	Opérandes autorisés
AND		0/1,%I,%Q,%M,%S,%X,%BLK.x,%•:Xk, [
ANDN		%I,%Q,%M,%S,%X,%BLK.x,%•:Xk, [
ANDR		%I
ANDF		%I

Chronogramme Le diagramme suivant illustre la temporisation des instructions AND.



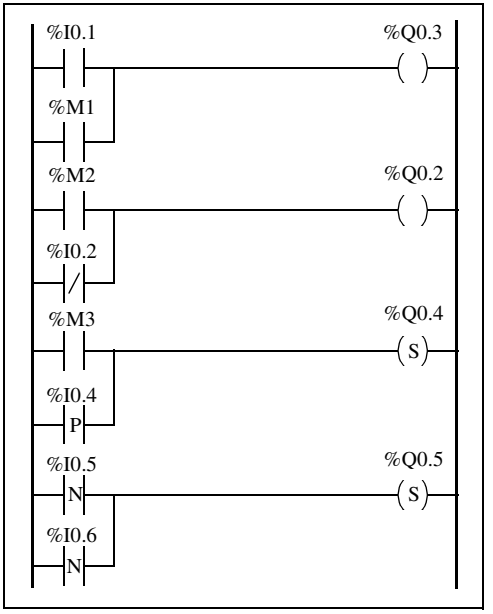
Instructions OR logique (OR, ORN, ORR, ORF)

Introduction

Les instructions OR effectuent une opération de liaison OR logique entre l'opérande (ou son inverse, ou son front montant ou descendant) et le résultat booléen de l'instruction précédente.

Exemples

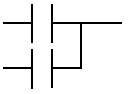
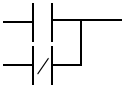
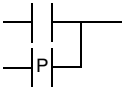
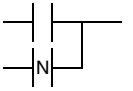
Les schémas suivants sont des exemples d'instructions OR.




```
LD      %I0.1
OR      %M1
ST      %Q0.3

LD      %M2
ORN     %I0.2
ST      %Q0.2

LD      %M3
ORR     %I0.4
S       %Q0.4

LDF     %I0.5
ORF     %I0.6
S       %Q0.5
```

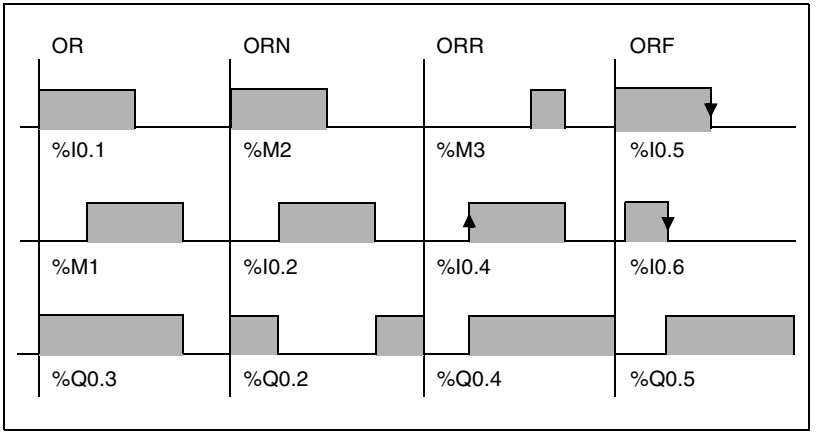
Opérandes autorisés

Le tableau suivant répertorie les types d'instructions OR, leurs équivalents dans le langage schéma à contacts, ainsi que les opérandes autorisés.

Instruction par liste	Symbole équivalent dans un schéma à contacts	Opérandes autorisés
OR		0/1,%I,%Q,%M,%S,%X,%BLK.x,%•:Xk
ORN		%I,%Q,%M,%S,%X,%BLK.x,%•:Xk
ORR		%I
ORF		%I

Chronogramme

Le diagramme suivant illustre la temporisation des instructions OR.



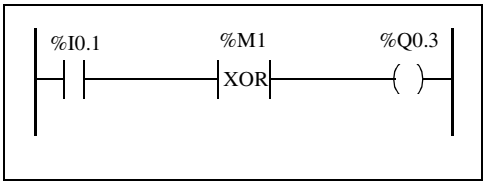
Instructions OR exclusif (XOR, XORN, XORR, XORF)

Introduction

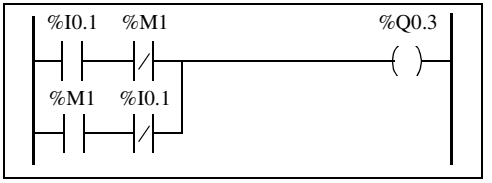
Les instructions XOR effectuent une opération de liaison OR exclusif entre l'opérande (ou son inverse, ou son front montant ou descendant) et le résultat booléen de l'instruction précédente.

Exemples

Les exemples suivants illustrent l'utilisation d'instructions XOR.




```
LD      %I0.1
XOR     %M1
ST      %Q0.3
```




```
LD      %I0.1
ANDN    %M1
OR(     %M1
ANDN    %I0.1
)
ST      %Q0.3
```

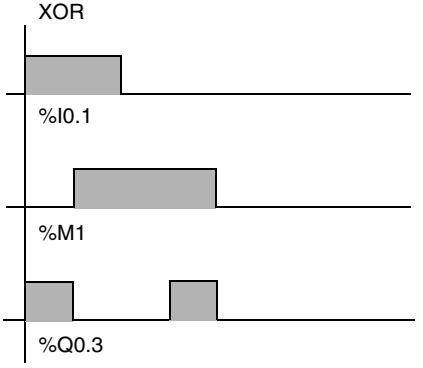
Opérandes autorisées

Le tableau suivant répertorie les types d'instructions XOR, ainsi que les opérandes autorisées.

Liste d'instructions	Opérandes autorisées
XOR	%I,%Q,%M,%S,%X,%BLK.x,%•:Xk
XORN	%I,%Q,%M,%S,%X,%BLK.x,%•:Xk
XORR	%I
XORF	%I

Chronogramme

Le diagramme suivant illustre la temporisation des instructions XOR.

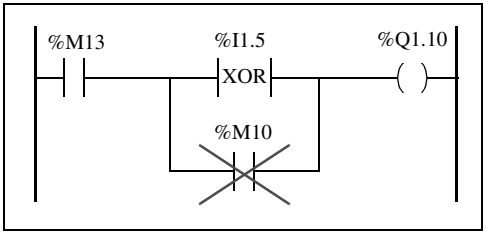


Cas spéciaux

Veuillez observer les précautions suivantes lors de l'utilisation d'instructions XOR dans des programmes en langage schéma à contacts :

- Ne commencez jamais un réseau par un contact XOR.
- N'insérez jamais de contacts XOR parallèlement à d'autres éléments du schéma à contacts (reportez-vous à l'exemple suivant.)

Comme l'illustre l'exemple suivant, l'insertion d'un élément parallèle à un contact XOR générera une erreur de validation.



Instruction NOT (N)

Introduction L'instruction NOT (N) inverse le résultat booléen de l'instruction précédente.

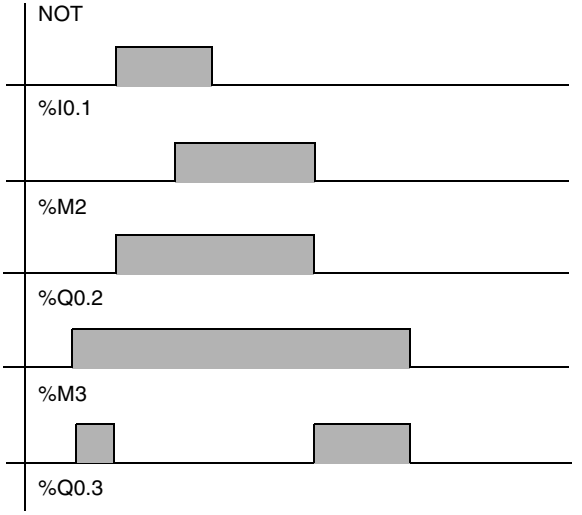
Exemple L'exemple suivant illustre l'utilisation de l'instruction NOT.

```
LD      %I0.1
OR      %M2
ST      %Q0.2
N
AND     %M3
ST      %Q0.3
```

Note : L'instruction NOT n'est pas réversible.

Opérandes autorisées Sans objet.

Chronogramme Le diagramme suivant illustre la temporisation de l'instruction NOT.



12.2 Blocs fonctions élémentaires

En bref...

Présentation

Cette rubrique présente des descriptions et des conseils de programmation relatifs aux blocs fonctions élémentaires.

Contenu de ce sous-chapitre

Ce sous-chapitre contient les sujets suivants :

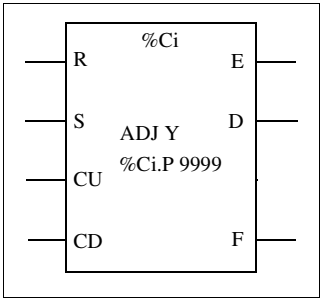
Blocs fonctions élémentaires

Introduction

Les blocs fonctions sont les sources des objets bits et des mots spécifiques utilisés par les programmes. Les blocs fonctions élémentaires comportent des fonctions simples telles que des temporisateurs ou des compteurs/décompteurs.

Exemple de bloc fonction

L'illustration suivante présente un exemple de bloc fonction compteur/décompteur.



Bloc compteur/décompteur

Objets bits

Les objets bits correspondent aux sorties blocs. Les instructions booléennes de test peuvent accéder à ces bits selon l'une ou l'autre de ces méthodes :

- directement (LD E, par exemple) s'ils sont liés au bloc par une programmation réversible (voir rubrique *Principes de programmation de blocs fonctions élémentaires, p. 224*).
- en spécifiant le type de bloc (LD %Ci.E, par exemple).

Les instructions peuvent accéder aux entrées.

Objets mots

Les objets mots correspondent aux paramètres et valeurs spécifiés suivants :

- **Paramètres de configuration du bloc :** Le programme peut accéder à certains paramètres (paramètres de pré-sélection, par exemple), mais pas à d'autres (base temps, par exemple).
- **Valeurs courantes :** %Ci.V, la valeur de comptage courante, par exemple.

Objets bits et objets mots accessibles

Le tableau suivant décrit les objets bits et les objets mots de blocs fonctions auxquels le programme a accès.

Bloc fonction élémentaire	Symbole	Plage (i)	Types d'objets	Description	Adresse	Accès en mode écriture
Temporisateur	%TMi	0 - 127	Mot	Valeur courante	%TMi.V	non
				Valeur de présélection	%TMi.P	oui
			Bit	Sortie du temporisateur	%TMi.Q	non
Compteur/ Décompteur	%Ci	0 - 31	Mot	Valeur courante	%Ci.V	non
				Valeur de présélection	%Ci.P	oui
			Bit	Sortie pour dépassement par valeur inférieure (vide)	%Ci.E	non
				Sortie prédéfinie atteinte	%Ci.D	non
				Sortie pour débordement (plein)	%Ci.F	non

Principes de programmation de blocs fonctions élémentaires

Introduction

Pour programmer des blocs de fonction élémentaires, appliquez l'une des méthodes suivantes :

- Utilisez des instructions de blocs fonctions (par exemple, `BLK` %TM2) : Cette méthode de programmation en langage schéma à contacts permet l'exécution d'opérations sur le bloc, à un emplacement unique du programme.
- Utilisez des instructions spécifiques (par exemple, `CU %Ci`) : Cette méthode non réversible permet l'exécution d'opérations sur les entrées du bloc, à plusieurs emplacements du programme (par exemple, `line 100 CU %C1`, `line 174 CD %C1`, `line 209 LD %C1.D`).

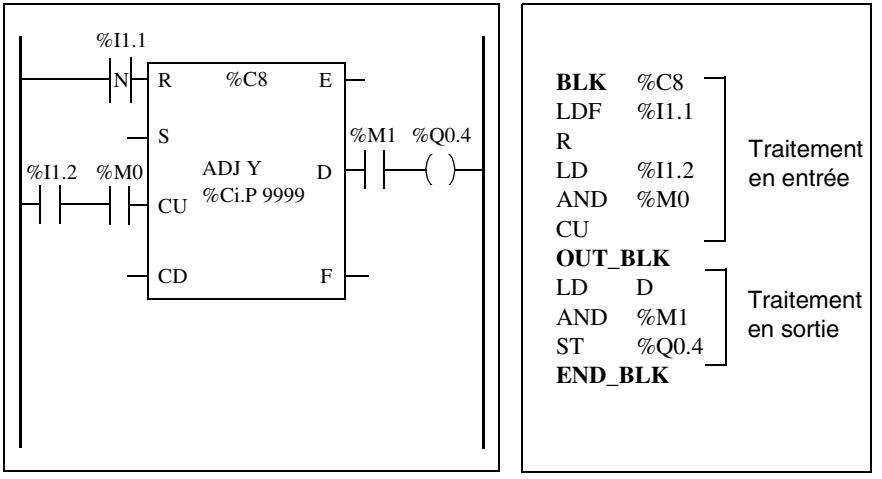
Programmation réversible

Utilisez les instructions BLK, OUT_BLK et END_BLK pour une programmation réversible :

- **BLK:** Indique le début du bloc.
- **OUT_BLK:** Utilisé pour lier directement les sorties du bloc.
- **END_BLK:** Indique la fin du bloc.

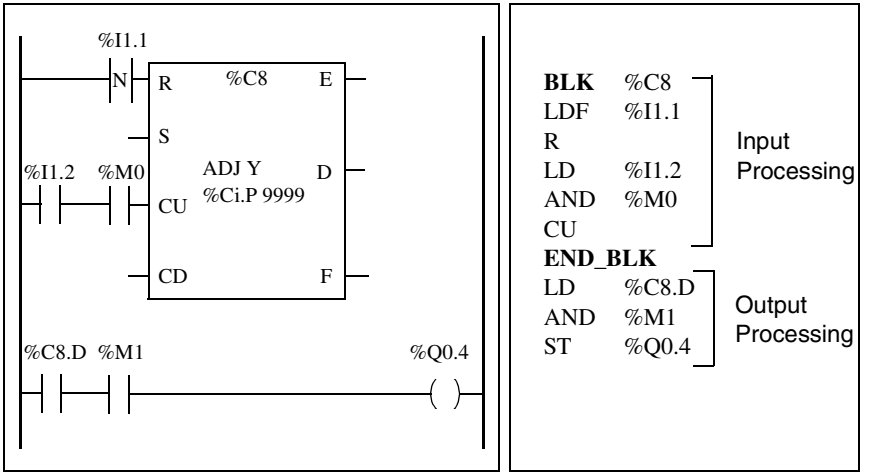
Exemple de sorties liées

Vous trouverez ci-dessous un exemple de programmation réversible d'un bloc fonction compteur avec des sorties liées.



Exemple sans sortie liée

Vous trouverez ci-dessous un exemple de programmation réversible d'un bloc fonction compteur dépourvu de sortie liée.



Note : Seules les instructions de test et d'entrée sur le bloc correspondant peuvent être placées entre les instructions BLK et OUT_BLK (ou entre BLK et END_BLK lorsque OUT_BLK n'est pas programmé).

Bloc fonction temporisateur (%TMi)

Introduction

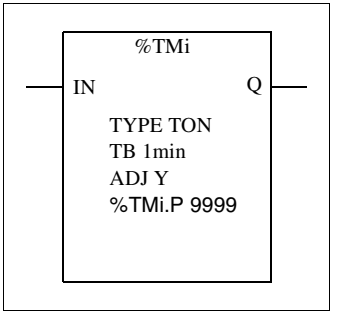
Il existe trois types de blocs fonctions temporisateur :

- TON (Timer On-Delay, temporisateur de délai à l'activation) : ce type de temporisateur permet de réguler les actions de délai à l'activation.
- TON (Timer Off-Delay, temporisateur de délai à la désactivation) : ce type de temporisateur permet de réguler les actions de délai à la désactivation.
- TP (Timer – Pulse, Temporisateur – Pulsation) : ce type de temporisateur permet de générer des pulsations d'une durée précise.

TwidoSoft permet de programmer et de modifier les délais de ces temporisateurs et/ou les durées des pulsations qu'ils génèrent.

Illustration

L'exemple suivant illustre l'utilisation du bloc fonction temporisateur.



Bloc fonction temporisateur

Paramètres

Le bloc fonction temporisateur possède les paramètres suivants :

Paramètre	Etiquette	Valeur
Numéro du temporisateur	%TMi	Automates compacts 0 à 63 Automates modulaires 0 à 127
Type	TON	• délai à l'activation (par défaut)
	TOF	• délai à la désactivation
	TP	• pulsation (monostable)
Base temps	TB	1 min (par défaut), 1 s, 100 ms, 10 ms, 1 ms (pour TM0 et TM1).
Valeur courante	%TMi.V	Mot avec incrémentation allant de 0 à %TMi.P lorsque le temporisateur est en cours d'exécution. Peut être lu et testé, mais pas écrit par le programme. %TMi.V peut être modifié par l'éditeur de données.
Valeur de présélection	%TMi.P	de 0 à 9999. Mot pouvant être lu, testé et écrit par le programme. La valeur par défaut est de 9999. La période ou le délai généré est égal à (%TMi.P x TB).
Editeur de données	Y/N	Y : Oui, la valeur %TMi.P de présélection peut être modifiée à l'aide de l'éditeur de données. N : Non, la valeur %TMi.P de présélection ne peut pas être modifiée.
Définition de l'entrée (ou de l'instruction)	IN	Démarre le temporisateur sur le front montant (types TON ou TP) ou descendant (type TOF).
Sortie du temporisateur	Q	Le bit associé %TMi.Q est réglé sur 1 en fonction de la fonction exécutée : TON, TOF ou TP.1.

Note : Plus la valeur de présélection est grande, plus le temporisateur sera précis.

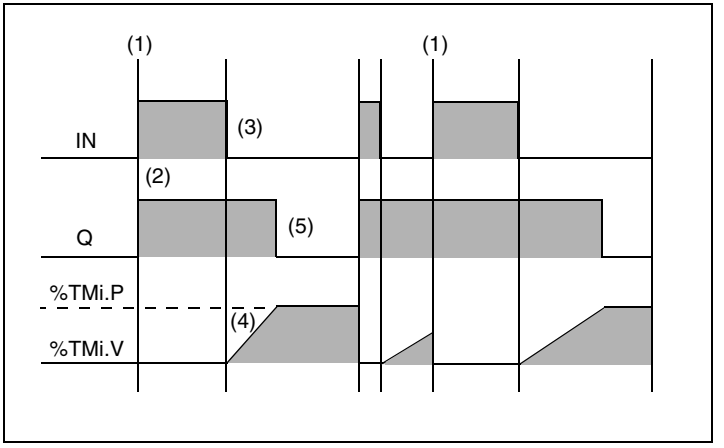
Type de temporisateur TOF

Introduction

Le type de temporisateur TOF (Timer Off-Delay, temporisateur de délai à la désactivation) permet de réguler les actions de délai à la désactivation. TwidoSoft permet de programmer ce délai.

Chronogramme

Le chronogramme suivant illustre le fonctionnement du type de temporisateur TOF.



Fonctionnement

Le tableau suivant décrit le fonctionnement du type de temporisateur TOF.

Phase	Description
1	La valeur courante %TMi.V est réglée sur 0 sur un front montant en entrée IN, et ce, même si le temporisateur est en cours d'exécution.
2	Le bit de sortie %TMi.Q est réglé sur 1 lorsqu'un front montant est détecté en entrée IN.
3	Le temporisateur démarre sur le front descendant de l'entrée IN.
4	La valeur courante %TMi.V augmente jusqu'à %TMi.P, par incréments d'une unité à chaque pulsation de la base temps TB.
5	Le bit de sortie %TMi.Q est remis à 0 lorsque la valeur courante atteint %TMi.P.

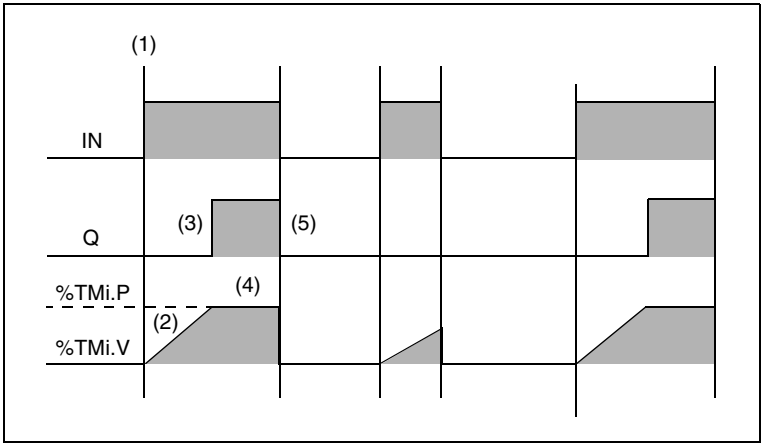
Type de temporisateur TON

Introduction

Le type de temporisateur TON (Timer On-Delay, temporisateur de délai à l'activation) permet de réguler les actions de délai à l'activation. TwidoSoft permet de programmer ce délai.

Chronogramme

Le chronogramme suivant illustre le fonctionnement du type de temporisateur TON.



Fonctionnement

Le tableau suivant décrit le fonctionnement du type de temporisateur TON.

Phase	Description
1	Le temporisateur démarre sur le front montant de l'entrée IN.
2	La valeur courante %TMi.V augmente de 0 à %TMi.P, par incréments d'une unité à chaque pulsation de la base temps TB.
3	Le bit de sortie %TMi.Q est réglé sur 1 lorsque la valeur courante a atteint %TMi.P.
4	Le bit de sortie %TMi.Q conserve la valeur 1 tant que la valeur de l'entrée IN est de 1.
5	Lorsqu'un front descendant est détecté en entrée IN, le temporisateur s'arrête, et ce, même s'il n'a pas atteint %TMi.P et que %TMi.V est réglé sur 0.

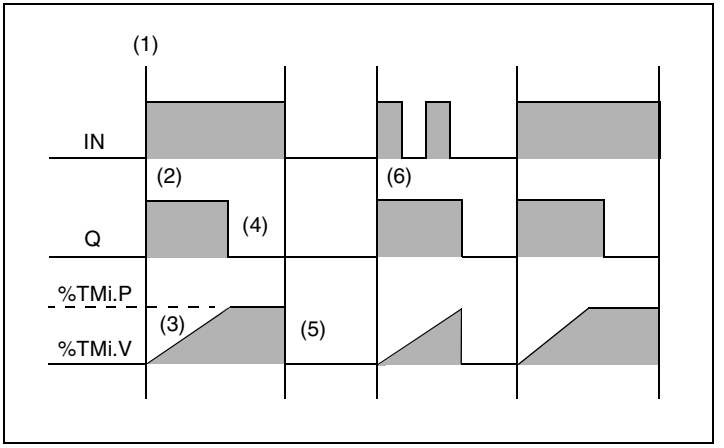
Type de temporisateur TP

Introduction

Le type de temporisateur TP (Timer – Pulse, Temporisateur – Pulsation) permet de générer des pulsations d'une durée spécifique. TwidoSoft permet de programmer ce délai.

Chronogramme

Le chronogramme suivant illustre le fonctionnement du type de temporisateur TP.



Fonctionnement

Le tableau suivant décrit le fonctionnement du type de temporisateur TP.

Phase	Description
1	Le temporisateur démarre sur le front montant de l'entrée IN. La valeur courante %TMi.V est réglée sur 0 si le temporisateur n'a pas encore démarré.
2	Le bit de sortie %TMi.Q est réglé sur 1 lorsque le temporisateur démarre.
3	La valeur courante %TMi.V du temporisateur augmente de 0 à %TMi.P, par incréments d'une unité à chaque pulsation de la base temps TB.
4	Le bit de sortie %TMi.Q est réglé sur 0 lorsque la valeur courante atteint %TMi.P.
5	La valeur courante %TMi.V est réglée sur 0 lorsque %TMi.V égale %TMi.P et que l'entrée IN retrouve la valeur 0.
6	Le temporisateur ne peut pas être remis à zéro. Lorsque %TMi.V égale %TMi.P et que l'entrée IN est réglée sur 0, %TMi.V est réglé sur 0.

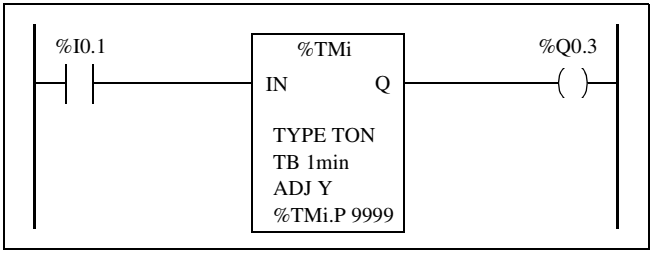
Programmation et configuration de temporisateurs

Introduction

Tous les blocs fonctions temporisateur (%TMi) sont programmés de la même façon, indépendamment de leur mode d'utilisation. La fonction temporisateur (TON, TOF ou TP) est sélectionnée au moment de la configuration.

Exemples

L'illustration suivante représente un bloc fonction temporisateur et affiche des exemples de programmation réversible et non réversible.



Programmation réversible

```
BLK     %TM1
LD      %I0.1
IN
OUT_BLK
LD      Q
ST      %Q0.3
END_BLK
```

Programmation non réversible

```
LD      %I0.1
IN      %TM1
LD      %TM1.Q
ST      %Q0.3
```

Configuration

Les paramètres suivants doivent être saisis au moment de la configuration :

- Type temporisateur : TON, TOF ou TP
- Base temps (TB) : 1 min, 1 s, 100 ms, 10 ms ou 1 ms
- Valeur de présélection (%TMi.P) : de 0 à 9999
- Réglage : Yes ou No (Y ou N)

Cas spéciaux

Le tableau suivant présente une liste des cas spéciaux de programmation et de configuration des temporisateurs.

Cas spécial	Description
Effet d'un redémarrage à froid (%S0=1)	Impose 0 à la valeur courante. Règle la sortie %TMi.Q sur 0. La valeur de présélection reprend la valeur réglée au moment de la configuration.
Effet d'un redémarrage à chaud (%S1=1)	Un redémarrage à chaud n'a aucun effet sur la valeur courante et la valeur de présélection du temporisateur. La valeur courante n'est pas modifiée lors d'une coupure secteur.
Effet d'un arrêt de l'automate	L'arrêt de l'automate ne provoque pas le gel de la valeur courante.
Effet d'un saut de programme	Le saut d'un bloc temporisateur ne provoque pas le gel du temporisateur. L'incrémentation du temporisateur se poursuit jusqu'à ce que la valeur de présélection (%TMi.P) soit atteinte. A ce stade, l'état du bit Terminé (%TMi.Q) affecté à la sortie Q du bloc temporisateur est modifié. Cependant, la sortie associée liée directement à la sortie bloc n'est ni activée, ni scrutée par l'automate.
Test par bit %TMi.Q (bit terminé)	Nous conseillons de ne tester le bit %TMi.Q qu'une seule fois dans le programme.
Effet de la modification de la valeur de présélection de %TMi.P	La modification de la valeur de présélection à l'aide d'une instruction ou d'un réglage ne prend effet qu'à la prochaine activation du temporisateur.

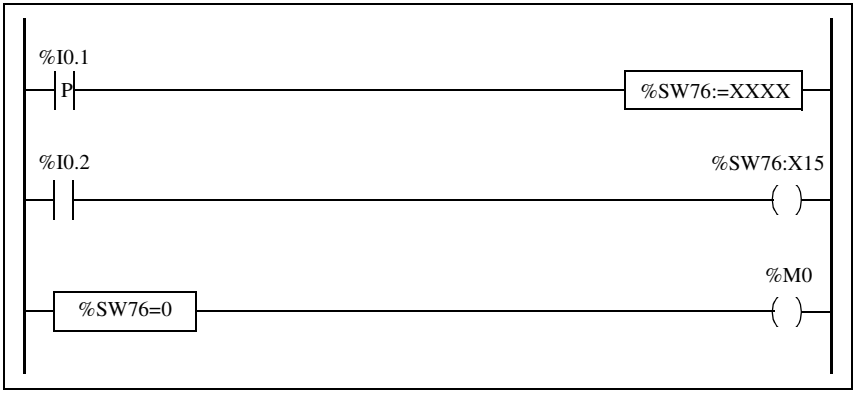
Temporisateurs avec base temps de 1 ms

La base temps de 1 ms n'est disponible que sur les temporisateurs %TM0 et %TM1. Les quatre mots système %SW76, %SW77, %SW78 et SW79 peuvent être utilisés comme des "sabliers". Les valeurs de ces quatre mots sont diminuées d'une unité par le système, toutes les millisecondes, **si ces mots ont une valeur positive.** Il est possible de créer une temporisation multiple en chargeant successivement un de ces mots ou en testant les valeurs intermédiaires. Les valeurs négatives de ces quatre mots ne seront pas modifiées. Un temporisateur peut être "gelé" en réglant le bit 15 sur la valeur 1, puis "dégelé" en remettant à zéro cette valeur.

Exemple de programmation

L'exemple suivant illustre la programmation d'un bloc fonction temporisateur.

```
LDR     %I0.1          (Lancement du temporisateur sur le front montant de
%I0.1)
[%SW76:=XXXX]          (XXXX = valeur requise)
LD      %I0.2          (gestion optionnelle du gel, gel de l'entrée I0.2)
ST      %SW76:X15
LD      [%SW76=0]      (réinitialisation du temporisateur de fin)
ST      %M0
.............
```



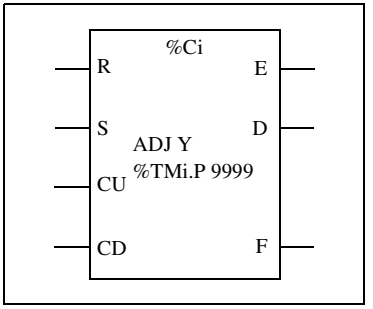
Bloc fonction compteur/décompteur (%Ci)

Introduction

Le bloc fonction compteur (%Ci) permet de compter ou de décompter des événements. Ces deux opérations peuvent être réalisées simultanément.

Illustration

L'illustration suivante présente un exemple de bloc fonction compteur/décompteur.



Bloc fonction compteur/décompteur

Paramètres

Le bloc fonction compteur possède les paramètres suivants :

Paramètre	Etiquette	Valeur
Numéro du compteur	%Ci	0 à 31
Valeur courante	%Ci.V	La valeur du mot est augmentée ou diminuée d'une unité en fonction des entrées (ou des instructions) CU et CD. Peut être lue et testée, mais pas écrite par le programme. Utilisez l'éditeur de données pour modifier %Ci.V.
Valeur de présélection	%Ci.P	0 - %Ci.P-9999. Le mot peut être lu, testé et écrit (valeur par défaut : 9999).
Edition à l'aide de l'éditeur de données	Y/N	• Y : Oui, la valeur de présélection peut être modifiée à l'aide de l'éditeur de données. • N : Non, la valeur de présélection ne peut pas être modifiée à l'aide de l'éditeur de données.
Entrée (ou instruction) de présélection	R	A l'état 1 : %Ci.V = 0.
Entrée (ou instruction) réglée	S	A l'état 1 : %Ci.V = %Ci.P.
Entrée (ou instruction) de compte croissant	CU	Augmente la valeur de %Ci.V d'une unité sur un front montant.
Entrée (ou instruction) de décompte (compte décroissant)	CD	Diminue la valeur de %Ci.V d'une unité sur un front montant.
Sortie pour dépassement par valeur inférieure	E (vide)	Le bit associé %Ci.E est égal à 1, lorsque la valeur du décompteur %Ci.V passe de 0 à 9999 (réglé sur 1 lorsque %Ci.V atteint 9999 et remis à zéro si le décomptage se poursuit).
Sortie prédéfinie atteinte	D (Terminé)	Le bit associé %Ci.D est égal à 1, lorsque %Ci.V est égal à %Ci.P.
Sortie pour débordement	F (plein)	Le bit associé %Ci.F est égal à 1, lorsque la valeur de %Ci.V passe de 9999 à 0 (réglé sur 1 lorsque %Ci.V atteint 0 et remis à zéro si le comptage croissant se poursuit).

Fonctionnement

Le tableau suivant décrit les étapes principales des opérations de comptage et de décomptage.

Fonctionnement	Action	Résultat
Compte croissant	Un bord montant apparaît sur le CU d'entrée de compte croissant (ou l'instruction CU est activée).	La valeur courante de %Ci.V est augmentée d'une unité.
	La valeur courante de %Ci.V est égale à la valeur de présélection de %Ci.P.	L'état du bit de sortie « préselection atteinte » %Ci.D affecté à la sortie D devient 1.
	La valeur courante de %Ci.V passe de 9999 à 0.	L'état du bit de sortie %Ci.F (dépassement de compte croissant) devient 1.
	Si le comptage se poursuit.	L'état du bit de sortie %Ci.F (dépassement de compte croissant) est remis à zéro.
Décompte (compte décroissant)	Un bord montant apparaît sur le CD d'entrée de décompte (ou l'instruction CD est activée).	La valeur courante de %Ci.V est diminuée d'une unité.
	La valeur courante de %Ci.V passe de 0 à 9999.	L'état du bit de sortie %Ci.E (dépassement par valeur inférieure) devient 1.
	Si le décomptage se poursuit.	L'état du bit de sortie %Ci.F (dépassement par valeur inférieure) est remis à zéro.
Compteur/ Décompteur	Pour utiliser simultanément les fonctions compteur et décompteur (ou pour activer les deux instructions CD et CU), les deux entrées CU et CD correspondantes doivent être régulées. Ces deux entrées sont ensuite scrutées. Si leur valeur est de 1, la valeur courante n'est pas modifiée.	
Remise à zéro	L'état de l'entrée R devient 1 (ou l'instruction R est activée).	Force la remise à zéro de la valeur %Ci.V. Les sorties %Ci.E, %Ci.D et %Ci.F sont réglées sur 0. L'entrée remise à zéro est prioritaire.
Définition	Si l'entrée S est réglée sur 1 (ou si l'instruction S est activée) et que l'entrée de présélection est réglée sur 0 (ou que l'instruction R est inactive).	La valeur courante %Ci.V prend la valeur de %Ci.P et la sortie %Ci.D est réglée sur 1.

Cas spéciaux

Le tableau suivant présente une liste des cas spéciaux de programmation et de configuration des compteurs.

Cas spécial	Description
Effet d'un démarrage à froid (%S0=1)	• La valeur courante de %Ci.V est réglée sur 0. • Les bits de sortie %Ci.E, %Ci.D et %Ci.F sont réglés sur 0. • La valeur de présélection est initialisée avec la valeur réglée au moment de la configuration
Effet d'une reprise à chaud (%S1=1) d'un arrêt de l'automate	N'a aucun effet sur la valeur courante du compteur (%Ci.V).
Effet de la modification de la valeur de présélection de %Ci.P	La modification de la valeur de présélection à l'aide d'une instruction ou d'un réglage ne prend effet qu'au moment du traitement du bloc par l'application (activation de l'une des entrées).

Programmation et configuration des compteurs

Introduction

L'exemple suivant illustre un compteur permettant de compter un maximum de 5 000 articles. Chaque impulsion sur l'entrée %I1.2 (lorsque le bit interne %M0 est réglé sur 1) incrémente la valeur du compteur %C8 d'une unité, jusqu'à la valeur de présélection finale (bit %C8.D=1). Le compteur est remis à zéro par l'entrée %I1.1.

Exemple de programmation

L'illustration suivante représente un bloc fonction compteur et affiche des exemples de programmation réversible et non réversible.

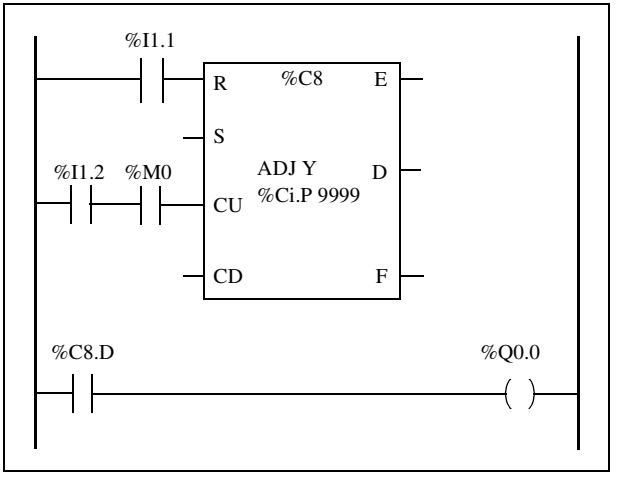


Schéma à contacts

```
BLK    %C8
LD     %I1.1
R
LD     %I1.2
AND    %M0
CU
END_BLK
LD     %C8.D
ST     %Q0.0
```

Programmation réversible

```
LD     %I1.1
R      %C8
LD     %I1.2
AND    %M0
CU     %C8
LD     %C8.D
ST     %Q0.0
```

Programmation non réversible

Configuration

Les paramètres suivants doivent être saisis au moment de la configuration :

- Valeur de présélection (%Ci.P) : réglée sur 5000 dans cet exemple
- Réglage : Oui

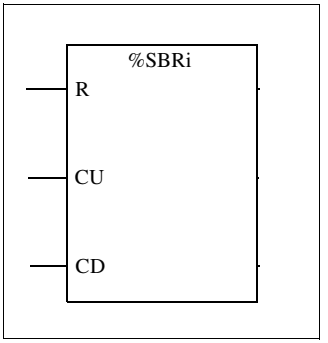
Bloc fonction registre bits à décalage (%SBRi)

Introduction

Le bloc fonction registre bits à décalage (%SBRi) effectue un décalage vers la gauche ou vers la droite des bits de données binaires (0 ou 1).

Illustration

L'exemple suivant illustre un bloc fonction registre à décalage :



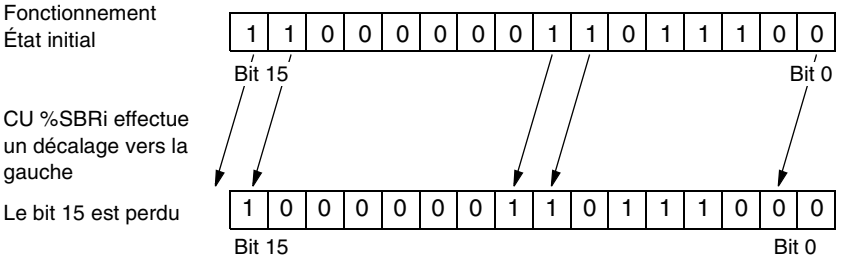
Paramètres

Le bloc fonction registre bits à décalage possède les paramètres suivants :

Paramètre	Étiquette	Valeur
Numéro de registre	%SBRi	0 à 7
Bit de registre	%SBRi.j	Les bits 0 à 15 (j = 0 à 15) du registre à décalage peut être testé par une instruction de test et écrit à l'aide d'une instruction d'affectation.
Entrée (ou instruction) RAZ	R	Sur un front montant, règle les bits de registre 0 à 15 %SBRi.j sur 0.
Décalage vers l'entrée (ou l'instruction) de gauche	CU	Sur un front montant, décale un bit de registre vers la gauche.
Décalage vers l'entrée (ou l'instruction) de droite	CD	Sur un front montant, décale un bit de registre vers la droite.

Fonctionnement

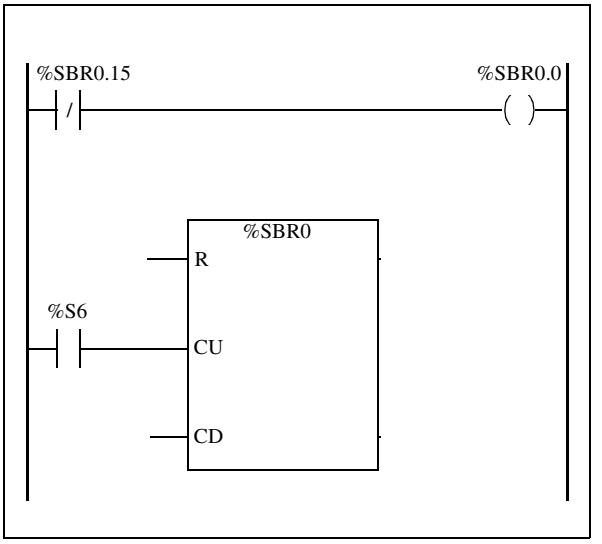
L'illustration suivante présente une configuration binaire avant et après une opération de décalage.



Cet exemple peut également s'appliquer à une requête de décalage d'un bit vers la droite (Bit 15 à Bit 0) à l'aide de l'instruction CD. Le bit 0 est perdu.
Si un registre de 16 bits n'est pas adapté, il est possible d'utiliser le programme pour afficher en cascade plusieurs registres.

Programmation

Dans l'exemple suivant, un bit est décalé vers la gauche à chaque seconde et le bit 0 prend l'état opposé au bit 15.



Programmation réversible

```
LDN    %SBR0.15
ST     %SBR0.0
BLK    %SBR0
LD     %S6
CU
END_BLK
```

Programmation non réversible

```
LDN    %SBR0.15
ST     %SBR0.0
LD     %S6
CU     %SBR0
```

Cas spéciaux

Le tableau suivant présente une liste des cas spéciaux de programmation du bloc fonction registre bits à décalage.

Cas spécial	Description
Effet d'un démarrage à froid (%S0=1)	Règle tous les bits du mot registre sur 0.
Effet d'une reprise à chaud (%S1=1)	N'a aucun effet sur les bits du mot registre.

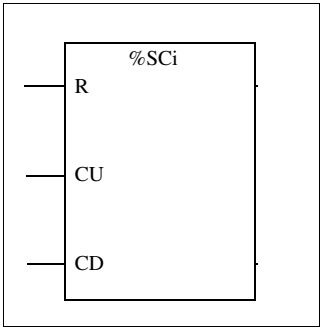
Bloc fonction pas à pas (%SCi)

Introduction

Un bloc fonction pas à pas (%SCi) permet d'accomplir une série d'étapes auxquelles des actions peuvent être affectées. Le passage d'une étape à l'autre dépend d'événements internes ou externes. Chaque fois qu'une étape est active, le bit associé est réglé sur 1. Une seule étape d'une fonction pas à peut être active à la fois.

Illustration

L'exemple suivant illustre un bloc fonction pas à pas :

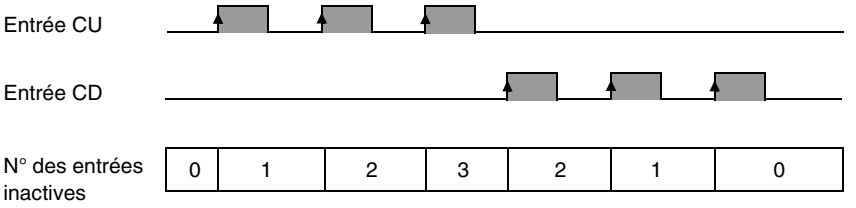


Paramètres

Le bloc fonction pas à pas possède les paramètres suivants :

Paramètre	Étiquette	Valeur
Numéro de fonction pas à pas	%SCi	0 à 7
Bit de fonction pas à pas	%SCi.j	Les bits de fonction pas à pas 0 à 255 (j = 0 à 255) peuvent être testés par une instruction logique de chargement et écrits à l'aide d'une instruction d'affectation.
Entrée (ou instruction) RAZ	R	Sur un front montant, réinitialise la fonction pas à pas.
Entrée (ou instruction) d'incrémentation	CU	Sur un front montant, incrémente la fonction pas à pas d'une étape.
Entrée (ou instruction) de décrémentation	CD	Sur un front montant, décrémente la fonction pas à pas d'une étape.

Chronogramme Le chronogramme suivant illustre le fonctionnement du bloc fonction pas à pas.

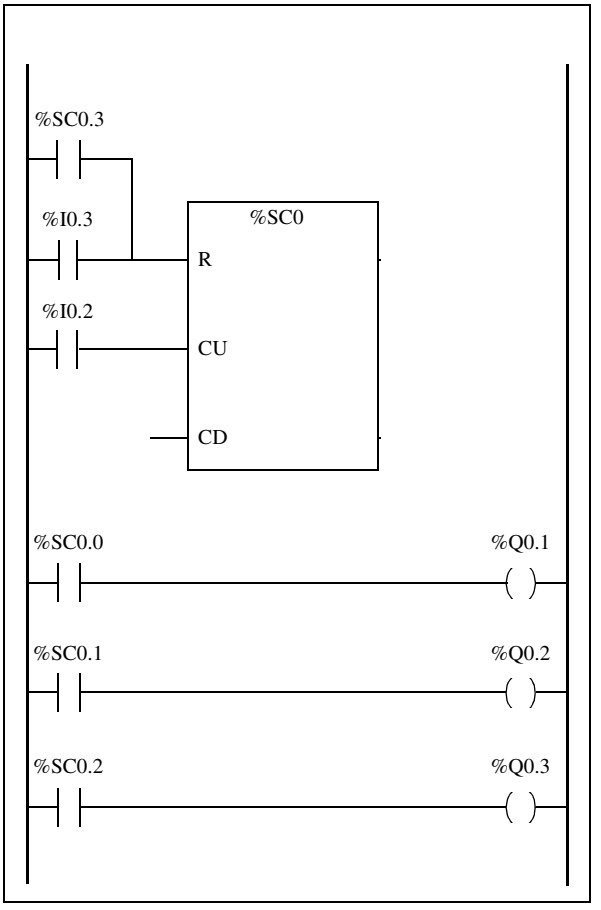


Programmation

L'exemple suivant illustre un bloc fonction pas à pas.

- La fonction pas à pas 0 est incrémentée d'une entrée %I0.2.
- La fonction pas à pas 0 est remise à 0 par l'entrée %I0.3 ou lorsqu'elle arrive à l'étape 3.
- L'étape 0 régule la sortie %Q0.1, l'étape 1 régule la sortie %Q0.2 et l'étape 2 régule la sortie %Q0.3.

L'illustration suivante présente la programmation réversible et non réversible correspondant à cet exemple.



Programmation réversible

```
BLK     %SC0
LD      %SC0.3
OR      %I0.3
R
LD      %I0.2
CU
END_BLK
LD      %SC0.0
ST      %Q0.1
LD      %SC0.1
ST      %Q0.2
LD      %SC0.2
ST      %Q0.3
```

Programmation non réversible

```
LD      %SC0.3
OR      %I0.3
R       %SC0
LD      %I0.2
CU      %SC0
LD      %SC0.0
ST      %Q0.1
LD      %SC0.1
ST      %Q0.2
LD      %SC0.2
ST      %Q0.3
```

Cas spéciaux

Le tableau suivant présente une liste des cas spéciaux de programmation du bloc fonction pas à pas.

Cas spécial	Description
Effet d'un démarrage à froid (%S0=1)	Initialise la fonction pas à pas.
Effet d'une reprise à chaud (%S1=1)	N'a aucun effet sur la fonction pas à pas.

12.3 Traitement numérique

Introduction au traitement numérique

Présentation

Cette rubrique offre une introduction au traitement numérique, qui s'appuie sur des descriptions et des directives de programmation.

Contenu de ce sous-chapitre

Ce sous-chapitre contient les sujets suivants :

Introduction aux instructions numériques

Présentation

Les instructions numériques s'appliquent généralement aux mots de 16 bits (voir section *Objets mots, p. 28*). Ces instructions apparaissent entre crochets. Si le résultat de l'opération logique précédente est True (accumulateur booléen = 1), l'instruction numérique est exécutée. Si ce résultat est False (accumulateur booléen = 0), l'instruction numérique n'est pas exécutée et l'opérande reste inchangé.

Instructions d'affectation

Introduction

Les instructions d'affectation permettent de charger l'opérande Op2 dans l'opérande Op1.

Affectation

Syntaxe des instructions d'affectation

[Op1:=Op2] <=> Op2 -> Op1

Les opérations d'affectation peuvent être exécutées sur :

- des chaînes de bits
- des mots
- des tables de mots

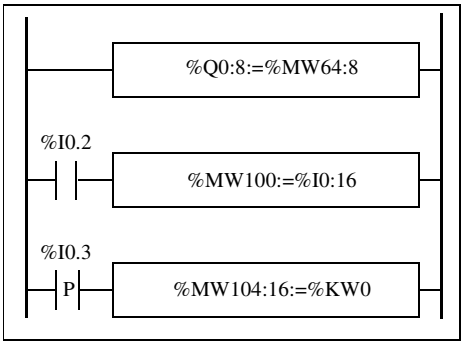
Affectation de chaînes de bits

Les opérations peuvent être exécutées sur les chaînes de bits suivantes (voir section *Objets structurés, p. 37*) :

- Chaîne de bit -> chaîne de bit (Exemple 1)
- Chaîne de bit -> mot (Exemple 2)
- Mot -> chaîne de bit (Exemple 3)
- Valeur immédiate -> chaîne de bit

Exemples

Exemples d'affectations de chaînes de bits



LD 1
[%Q0:8:=%M64:8] (Ex. 1)

LD %I0.2
[%MW100:=%I0:16] (Ex. 2)

LDR %I0.3
[%M104:16:=%KW0] (Ex. 3)

Règles d'utilisation :

- Pour l'affectation chaîne de bits -> mot : les bits de la chaîne sont transférés vers le mot en commençant par la droite (premier bit de la chaîne vers bit 0 du mot) et les bits de mot non concernés par le transfert (longueur<16) sont réglés sur 0.
- Pour l'affectation mot -> chaîne de bits : les bits de mot sont transférés en partant de la droite (bit de mot 0 vers premier bit de la chaîne).

Affectations de chaînes de bits

Syntaxe des affectations de chaînes de bits

Opérateur	Syntaxe	Opérande 1 (Op1)	Opérande 2 (Op2)
:=	[Op1: = Op2] L'opérande 1 (Op1) prend la valeur de l'opérande 2 (Op2).	%MWi,%QWi, %SWi %MWi[MWi], **%Mi:L, %Qi:L, %Si:L, %Xi:L**	Valeur immédiate, %MWi, %KWi, %IW, %INWi, %QW, %QNWi, %SWi, %BLK.x, %MWi[MWi], %KWi[MWi], **%Mi:L,%Qi:L, %Si:L, %Xi:L, %Ii:L**

Note : L'abréviation %BLK.x (%C0.P, par exemple) est utilisée pour décrire tout mot de bloc fonction.

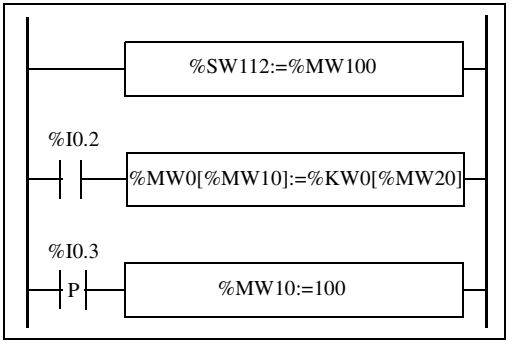
Affectation de mots

Les opérations d'affectation peuvent être exécutées sur les mots suivants :

- Mot -> mot (Exemple 1)
- Mot indexé -> mot
- Valeur immédiate -> mot (Exemple 3)
- Chaîne de bit -> mot
- Mot -> mot indexé
- Mot indexé -> mot indexé (Exemple 2)
- Valeur immédiate -> mot indexé
- Mot -> chaîne de bit

Exemples

Exemples d'affectations de mots




```
LD      1
[%SW112:=%MW100]      (Ex. 1)

LD      %I0.2
[%MW0[%MW10]:=]       (Ex. 2)
%KW0[%MW20]

LDR     %I0.3         (Ex. 3)
[%MW10:=100]
```

Syntaxe

Syntaxe des affectations de mots

Opérateur	Syntaxe	Opérande 1 (Op1)	Opérande 2 (Op2)
:=	[Op1: = Op2] L'opérande 1 (Op1) prend la valeur de l'opérande 2 (Op2).	**%BLK.x, %MWi, %QWi, %SWi %MWi[MWi]**, %Mi:L, %Qi:L, %Si:L, %Xi:L	**Valeur immédiate, %MWi, %KWi, %IW, %QW, %SWi, %MWi[MWi], %KWi[MWi]**, %INW, %Mi:L, %Qi:L, %QNW, %Si:L, %Xi:L, %Ii:L

Note : L'abréviation %BLK.x (%R3.I, par exemple) est utilisée pour décrire tout mot de bloc fonction. Pour les chaînes de bits %Mi:L, %Si:L et %Xi:L, le repère de base du premier bit de la chaîne doit être un multiple de 8 (0, 8, 16, ..., 96, ...).

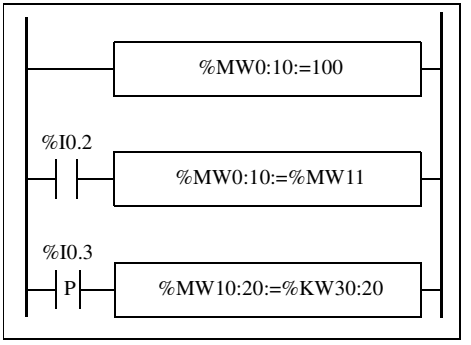
Affectation de tables de mots

Les opérations d'affectation peuvent être exécutées sur les tables de mots suivantes (voir section *Tables de mots, p. 38*) :

- Valeur immédiate -> table de mot (Exemple 1)
- Mot -> table de mot (Exemple 2)
- Table de mot -> table de mot (Exemple 3)
  La longueur de la table (L) doit être la même pour les deux tables.

Exemples

Exemples d'affectations de tables de mots




```
LD      1
[%MW0:10:=100]                (Ex. 1)

LD      %I0.2
[%MW0:10:=%MW11]              (Ex. 2)

LDR     %I0.3
[%MW10:20:=%KW30:20]          (Ex. 3)
```

Syntaxe

Syntaxe des affectations de tables de mots

Opérateur	Syntaxe	Opérande 1 (Op1)	Opérande 2 (Op2)
:=	[Op1: = Op2] L'opérande 1 (Op1) prend la valeur de l'opérande 2 (Op2).	**%MWi:L, %SWi:L**	**%MWi:L, %SWi:L**, Valeur immédiate, %MWi, %KWi, %IW, %QW, %SWi, %BLK.x

Note : L'abréviation %BLK.x (%R3.I, par exemple) est utilisée pour décrire tout mot de bloc fonction. Pour les chaînes de bits %Mi:L, %Si:L et %Xi:L, le repère de base du premier bit de la chaîne doit être un multiple de 8 (0, 8, 16, ..., 96, ...).

Instructions de comparaison

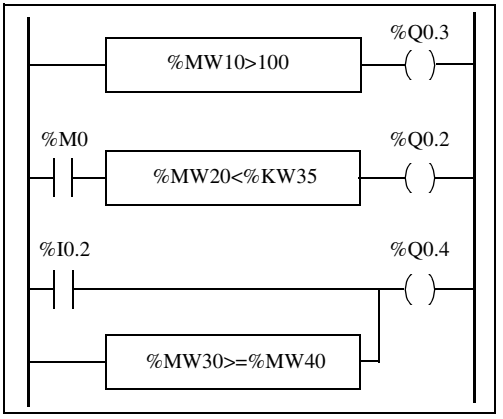
Introduction

Les instructions de comparaison permettent de comparer deux opérandes.
Le tableau suivant répertorie les différents types d'instructions de comparaison.

Instruction	Fonction
>	Teste si l'opérande 1 est supérieur à l'opérande 2.
>=	Teste si l'opérande 1 est supérieur ou égale à l'opérande 2.
<	Teste si l'opérande 1 est inférieur à l'opérande 2.
<=	Teste si l'opérande 1 est inférieur ou égal à l'opérande 2.
=	Teste si l'opérande 1 est égal à l'opérande 2.
<>	Teste si l'opérande 1 est différent de l'opérande 2.

Structure

La comparaison s'effectue entre les crochets qui suivent les instructions LD, AND et OR. Le résultat est 1 lorsque le résultat de la comparaison requise est True.
Exemples d'instructions de comparaison




```
LD    [%MW10 > 100]
ST    %Q0.3

LD    %M0
AND   [%MW20 < %KW35]
ST    %Q0.2

LD    %I0.2
OR    [%MW30>=%MW40]
ST    %Q0.4
```

Syntaxe

Syntaxe des instructions de comparaison

Opérateur	Syntaxe	Opérande 1 (Op1)	Opérande 2 (Op2)
>, >=, <, <=, =, <>	LD [Op1 Opérateur Op2] AND [Op1 Opérateur Op2] OR [Op1 Opérateur Op2]	%MWi, %KWi, %INWi, %IW, %QNWi, %QWi, %QNWi, %SWi, %BLK.x	Valeur immédiate, %MWi, %KWi, %INWi, %IW, %QNWi, %QW, %SWi, %BLK.x, %MWi [%MWi], %KWi [%MWi]

Note : Les instructions de comparaison peuvent apparaître entre parenthèses.

Exemple d'utilisation d'une instruction de comparaison entre parenthèses

```
LD      %M0
AND(    [%MW20 > 10]
OR      %I0.0
)
ST      %Q0.1
```

Instructions arithmétiques

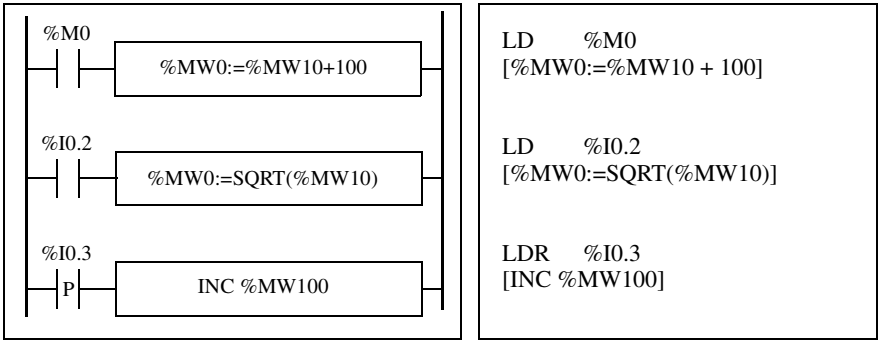
Introduction

Les instructions arithmétiques permettent d'effectuer des opérations arithmétiques entre deux opérandes ou sur une opérande.
Le tableau suivant répertorie les différents types d'instructions arithmétiques.

Instruction	Fonction
+	Addition de deux opérandes
-	Soustraction de deux opérandes
*	Multiplication de deux opérandes
/	Division de deux opérandes
REM	Reste de la division de deux opérandes
SQRT	Racine carrée d'une opérande
INC	Incrémentation d'une opérande
DEC	Décrémentation d'une opérande

Structure

Les opérations arithmétiques sont effectuées de la façon suivante :



Syntaxe

La syntaxe dépend des opérateurs utilisés, tel que l'indique le tableau ci-dessous.

Opérateur	**Syntaxe**	**Opérande 1 (Op1)**	**Opérandes 2 et 3 (Op2 & 3)**
+, -, *, /, REM	[Op1: = Op 2 Opérateur Op3]	%MWi, %QWi, %SWi	Valeur immédiate (2), %MWi, %KWi, %INW, %IW, %QNW, %QW, %SWi, %BLK.x
SQRT (1)	[Op1: = SQRT(Op2)]		
INC, DEC	[Opérateur Op1]		

Note : (1) Avec SQRT, Op2 ne peut pas être une valeur immédiate.

Débordement et conditions d'erreurs

Addition

- Débordement pendant l'opération
  Si le résultat dépasse les limites de -32768 ou de +32767, le bit %S18 (débordement) est réglé sur 1. Le résultat est alors incorrect en soi (voir Exemple 1 page suivante). Le programme utilisateur gère le bit %S18.
- Débordement absolu du résultat (arithmétique sans signe)
  Lors de certains calculs, il peut être nécessaire d'interpréter une opérande au moyen de l'arithmétique sans signe (le bit 15 représente alors la valeur 32768). La valeur maximum d'une opérande est 65535. L'addition de deux valeurs absolues (sans signe) dont le résultat est supérieur à 65535 provoque un débordement. Ceci est signalé par l'affectation de la valeur 1 au bit système %S17 (retenue), qui représente la valeur 65536.

Soustraction

- Résultat négatif
  Si le résultat de la soustraction est négatif, le bit système %S17 est réglé sur 1.

Multiplication

- Débordement pendant l'opération
  Si le résultat dépasse la capacité du mot de résultat, le bit %S18 (débordement) est réglé sur 1 et le résultat n'est pas significatif.

Division / reste

- Division par 0
  Si le dividende est 0, la division est impossible et le bit système %S18 est réglé sur 1. Le résultat est alors incorrect.
- Débordement pendant l'opération
  Si le quotient de la division dépasse la capacité du mot de résultat, le bit système %S18 est réglé sur 1.

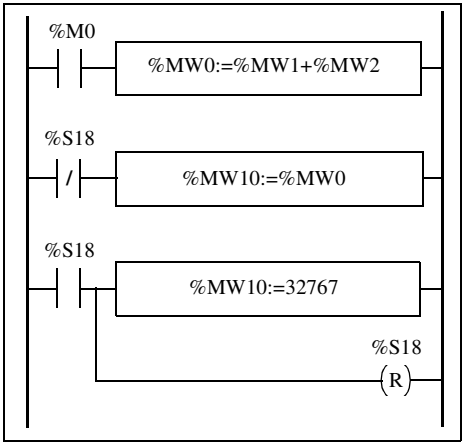
Calcul de la racine carrée

- Débordement pendant l'opération
  La calcul de la racine carrée est uniquement effectué sur les valeurs positives. Le résultat est, par conséquent, toujours positif. Si l'opérande de racine carrée est négative, le bit système %S18 est réglé sur 1 et le résultat est incorrect.

Note : Le programme utilisateur gère les bits système %S17 et %S18. L'automate les règle sur 1. Ils doivent être remis à 0 par le programme afin de pouvoir être réutilisés (voir exemple page précédente).

Exemples

Exemple 1 : débordement lors de l'addition




```
LD      %M0
[%MW0:=%MW1 + %MW2]

LDN     %S18
[%MW10:=%MW0]

LD      %S18
[%MW10:=32767]
R       %S18
```

Si %MW1 =23241 et %MW2=21853, le résultat réel (45094) ne peut pas être exprimé par un mot de 16 bits, le bit %S18 est réglé sur 1 et le résultat obtenu (-20442) est incorrect. Dans cet exemple, la valeur est fixée à 32767 lorsque le résultat est supérieur à cette valeur.

Exemple 2 : [%MW2:=%MW0 + %MW1], où %MW0 =65086, %MW1=65333. Le mot %MW2 contient le nombre 64883. Le bit %S17 est réglé sur 1 et représente la valeur 65536. Le résultat arithmétique sans signe est alors égal à : 65536 + 64883 = 130419.

Exemple 3 : [%MW2:=%MW0 + %MW1], où %MW0 =45736 (soit une valeur avec signe de -19800), %MW1=38336 (soit une valeur avec signe de 27200). Les deux bits système %S17 et %S18 sont réglés sur 1. Le résultat arithmétique avec signe (+18536) est incorrect. En arithmétique sans signe, le résultat (18536 + la valeur de %S17, qui est 84072) est correct.

Instructions logiques

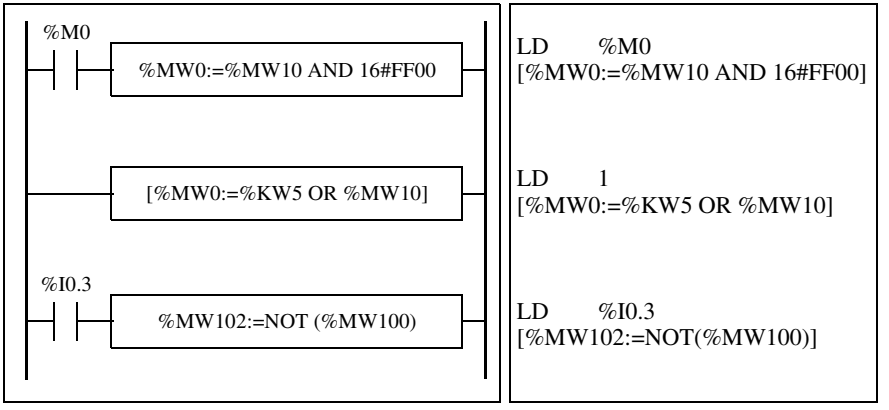
Introduction

Les instructions logiques permettent d'effectuer des opérations logiques entre deux opérandes par mot ou sur un opérande par mot.
Le tableau suivant répertorie les différents types d'instructions logiques :

Instruction	Fonction
AND	AND (bit à bit) entre deux opérandes
OR	OR logique (bit à bit) entre deux opérandes
XOR	OR exclusif (bit à bit) entre deux opérandes
NOT	Complément logique (bit à bit) d'un opérande

Structure

Les opérations logiques sont effectuées de la façon suivante :



Syntaxe

La syntaxe dépend des opérateurs utilisés :

Opérateur	Syntaxe	Opérande 1 (Op1)	Opérandes 2 et 3 (Op2 et 3)
AND, OR, XOR	[Op1: = Op 2 Opérateur Op3]	%MWi, %QWi, %SWi	Valeur immédiate (1), %MWi, %KWi, %IW, %QW, %SWi, %BLK.x
NOT	[NOT(Op2)]		

Note : (1) Avec NOT, Op2 ne peut pas être une valeur immédiate.

Exemple

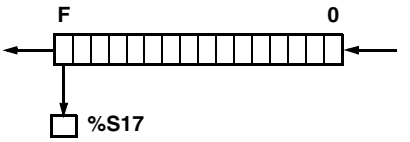
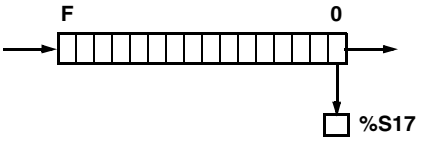
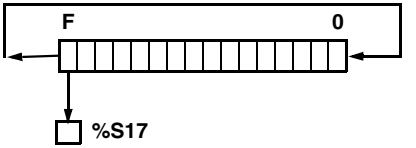
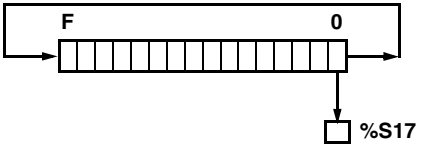
L'exemple suivant présente une instruction AND logique.

```
[%MW15:=%MW32 AND %MW12]
```

Instructions de décalage

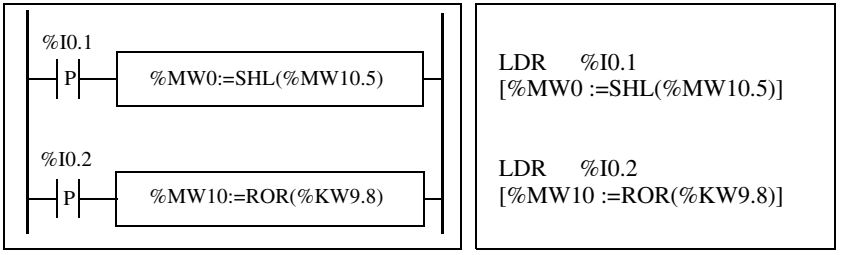
Introduction

Les instructions de décalage déplacent les bits d'un opérande d'un certain nombre de positions vers la droite ou vers la gauche.
Le tableau suivant répertorie les différents types d'instructions de décalage.

Instruction	Fonction	
Décalage logique		
SHL(op2,i)	Décalage logique de i positions vers la gauche	
SHR(op2,i)	Décalage logique de i positions vers la droite	
Décalage circulaire		
ROL(op2,i)	Décalage circulaire de i positions vers la gauche	
ROL(op2,i)	Décalage circulaire de i positions vers la droite	

Structure

Les opérations de décalage sont effectuées de la façon suivante :



Syntaxe

La syntaxe dépend des opérateurs utilisés, comme indiqué dans le tableau suivant.

Opérateur	Syntaxe	Opérande 1 (Op1)	Opérande 2 (Op2)
SHL, SHR	[Op1: = Opérateur (Op2,i)]	%MWi, %QWi, %SWi	%MWi, %KWi, %IW, %QW, %SWi, %BLK.x
ROL, ROR			

Instructions de conversion

Introduction

Les instructions de conversion permettent d'effectuer la conversion entre les différentes représentations numériques.
Le tableau suivant répertorie les différents types d'instructions de conversion.

Instruction	Fonction
BTI	BCD --> Conversion binaire
ITB	Binaire --> Conversion BCD

Révision du code BCD

Le codage BCD (Binary Coded Decimal - décimal codé binaire) représente les décimaux (entre 0 et 9) par un code à quatre bits binaires. Un objet mot de 16 bits peut ainsi contenir un numéro exprimé par quatre chiffres (0000 - 9999).
Lors d'une conversion, le bit système %S18 est réglé sur 1 si la valeur n'est pas BCD. Ce bit doit être testé et remis à 0 par le programme.
Représentation BCD des décimaux :

Décimal	0	1	2	3	4	5	6	7	8	9
BCD	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001

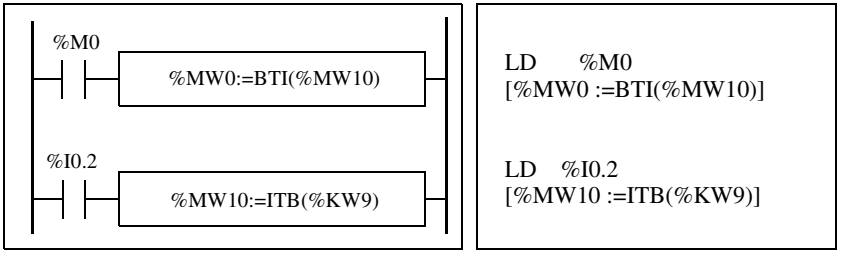
Exemples :

- Le mot %MW5 exprime la valeur BCD « 2450 », qui correspond à la valeur binaire 0010 0100 0101 0000
- Le mot %MW12 exprime la valeur décimale « 2450 », qui correspond à la valeur binaire 0000 1001 1001 0010.

Le mot %MW5 est converti en mot %MW12 à l'aide de l'instruction BTI.
Le mot %MW12 est converti en mot %MW5 à l'aide de l'instruction ITB.

Structure

Les opérations de conversion sont effectuées de la façon suivante :



Syntaxe

La syntaxe dépend des opérateurs utilisés, tel que l'indique le tableau ci-dessous.

Opérateur	Syntaxe	Opérande 1 (Op1)	Opérande 2 (Op2)
BTI, ITB	[Op1: = Opérateur (Op2,i)]	%MWi, %QWi, %SWi	%MWi, %KWi, %IW, %QW, %SWi, %BLK.x

Exemples d'application :

L'instruction BTI peut être utilisée pour traiter une valeur de consigne aux entrées de l'automate via des molettes codées en BCD.
L'instruction peut être utilisée pour afficher des valeurs numériques sur des écrans codés en BCD (résultat d'un calcul, valeur courante d'un bloc fonction, par exemple).

12.4 Instructions sur programme

Introduction aux instructions sur programme

Présentation

Cette rubrique présente une introduction aux instructions sur programme.

Contenu de ce sous-chapitre

Ce sous-chapitre contient les sujets suivants :

Instructions END

Introduction

Les instructions END définissent la fin de l'exécution de la scrutation d'un programme.

END, ENDC et ENDCN

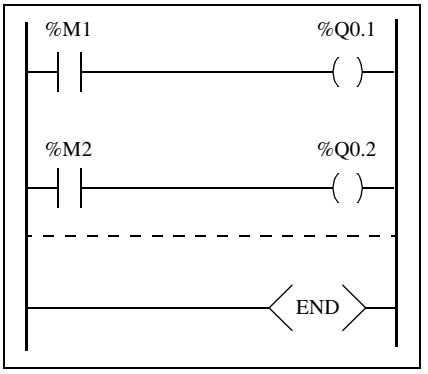
Il existe trois instructions END différentes :

- END : fin de programme inconditionnelle.
- ENDC : fin de programme si le résultat booléen de l'instruction de test précédente est 1.
- ENDCN : fin de programme si le résultat booléen de l'instruction de test précédente est 0.

Par défaut (en mode Normal), des sorties sont générées et la scrutation suivante est lancée dès la fin d'un programme.
Si la scrutation est périodique, des sorties sont générées et la scrutation suivante est lancée dès que la fin de période est atteinte.

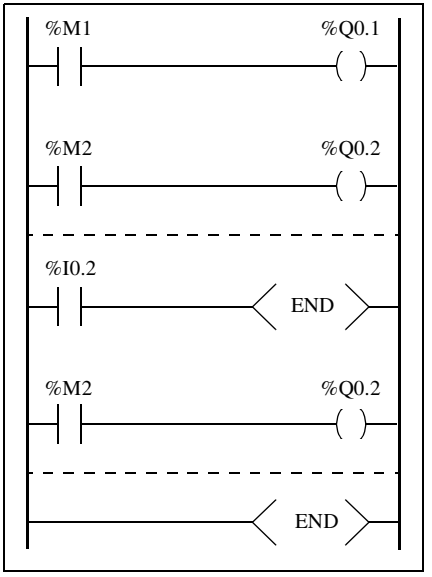
Exemples

Exemple d'instruction END inconditionnelle




```
LD      %M1
ST      %Q0.1
LD      %M2
ST      %Q0.2

.....

END
```

Exemple d'instruction END conditionnelle




```
LD      %M1
ST      %Q0.1
LD      %M2
ST      %Q0.2

.....

LD      %I0.2
ENDC                  → If %I0.2 = 1, end of program scanning
LD      %M2
ST      %Q0.2

                      If %I0.2 = 0, continues program scanning until new END instruction
.....

END
```

Instruction NOP

NOP	L'instruction NOP n'effectue aucune opération. Utilisez cette instruction pour « réserver » des lignes d'un programme afin de pouvoir insérer ultérieurement des instructions, sans modifier les numéros de ligne.

Instructions de saut

Introduction

Les instructions de saut ont pour effet d'interrompre immédiatement l'exécution d'un programme et de le reprendre à partir de la ligne suivant la ligne contenant l'étiquette %Li (i = 0 à 15).

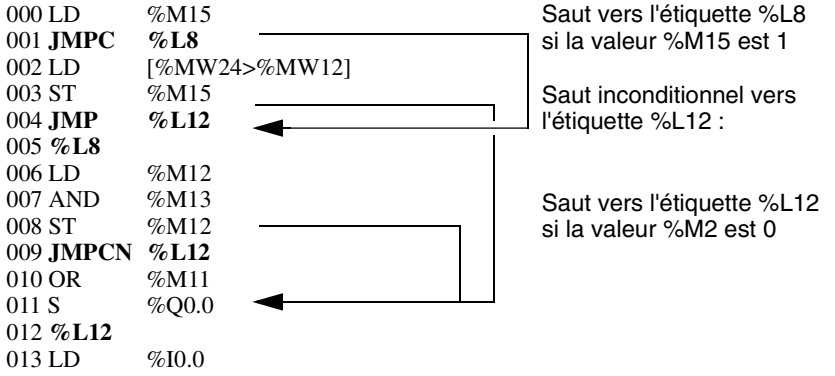
JMP, JMPC et JMPCN

Trois instructions de saut différentes sont disponibles :

- JMP : saut de programme inconditionnel
- JMPC : saut de programme si le résultat booléen de la logique précédente est 1.
- JMPCN : saut de programme si le résultat booléen de la logique précédente est 0.

Exemples

Exemples d'instructions de saut



Directives

- Les instructions de saut ne peuvent pas apparaître entre parenthèses et ne doivent pas être placées entre les instructions AND(, OR(et une parenthèse fermante «)».
- L'étiquette peut uniquement être placée devant une instruction LD, LDN, LDR, LDF ou BLK.
- Le numéro de l'étiquette %Li doit être défini une seule fois dans un programme.
- Le saut de programme est effectué vers une ligne de programmation en amont ou en aval. Lorsque le saut est en amont, le temps de scrutation doit être contrôlé. Un temps de scrutation trop long peut provoquer l'expiration du temporisateur chien de garde.

Instructions de sous-programme

Introduction

Les instructions de sous-programme déclenchent l'exécution d'un sous-programme, puis le retour vers le programme principal.

SRn, SRn: et RET

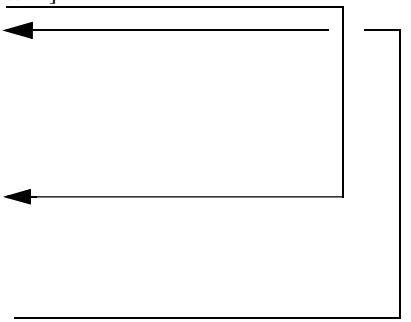
Les sous-programmes se composent de trois étapes :

- L'instruction **SRn** appelle le sous-programme référencé par l'étiquette SRn, si le résultat de l'instruction booléenne précédente est 1.
- Le sous-programme est référencé par l'étiquette **SRn:**, n pouvant prendre une valeur comprise entre 0 à 15 pour TWDLCAA10DRF, TWDLCAA16DRF et 0 à 63 pour tous les autres automates.
- L'instruction **RET** placée à la fin du sous-programme provoque le retour au programme principal.

Exemple

Exemples d'instructions de sous-programme

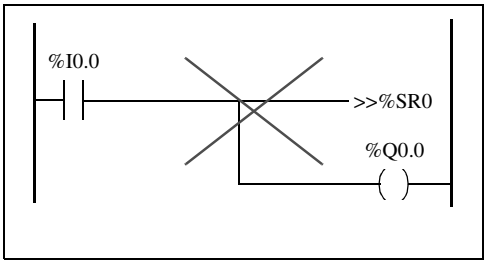
```
000  LD      %M15
001  AND     %M5
002  ST      %Q0.0
003  LD      [%MW24>%MW12]
004  SR8
005  LD      %I0.4
006  AND     M13
007  _
008  _
009  _
010  END

011  SR8:
012  LD      1
013  IN      %TM0
014  LD      [%TM0.Q]
015  ST      %M15
010  RET
```

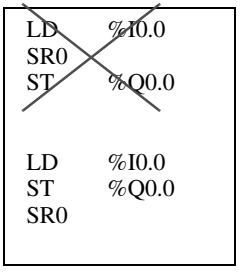


Directives

- Un sous-programme ne doit pas appeler un autre sous-programme.
- Les instructions de sous-programme ne peuvent pas apparaître entre parenthèses et ne doivent pas être placées entre les instructions AND(, OR(et une parenthèse fermante ")".
- L'étiquette peut uniquement être placée devant une instruction LD ou BLK pour marquer le début d'une équation booléenne (ou d'un réseau booléen).
- L'appel du sous-programme ne doit pas être suivi d'une instruction d'affectation. En effet, le sous-programme risque de modifier le contenu de l'accumulateur booléen. Aussi celui risque d'avoir une valeur de retour différente de celle qu'il avait avant l'appel. Voir l'exemple suivant.

Exemple de programmation d'un sous-programme





Instructions avancées

13

En bref...

Présentation

Ce chapitre offre des informations sur les instructions et les blocs fonctions utilisés pour créer des programmes de régulation avancés destinés aux automates Twido.

Contenu de ce chapitre

Ce chapitre contient les sous-chapitres suivants :

13.1 Blocs fonctions avancés

En bref...

Présentation

Cette rubrique offre une présentation des blocs fonctions avancés et contient des exemples de programmation.

Contenu de ce sous-chapitre

Ce sous-chapitre contient les sujets suivants :

Objets mots et objets bits associés à des blocs fonctions avancés

Introduction

Les blocs fonctions avancés utilisent des mots et des bits dédiés de type identique comme blocs fonctions élémentaires. L'utilisation de blocs fonctions avancés est réservée aux programmateurs les plus expérimentés. Les blocs fonctions avancés comprennent :

- les registres LIFO/FIFO (%R) ;
- les programmateurs cycliques (%DR) ;
- les compteurs rapides (%FC) ;
- les compteurs rapides (%VFC) ;
- la sortie de modulation de la largeur de l'impulsion (%PWM) ;
- la sortie du générateur d'impulsion (%PLS) ;
- le registre bits à décalage (%SBR) ;
- le compteur à décalage (%SC) ;
- le bloc contrôle message (%MSG).

Objets accessibles par le programme

Le tableau suivant présente les mots et les bits associés aux différents blocs fonctions avancés. Veuillez noter que l'accès en écriture mentionné dans le tableau suivant dépend du paramètre « Réglable », sélectionné au moment de la configuration. Ce réglage permet d'autoriser ou de refuser l'accès aux mots ou aux bits par TwidoSoft ou par l'interface opérateur.

Bloc fonction avancé	Mots et bits associés		Adresse	Accès en écriture
%R	Mot	Accès au registre	%Ri.I	Oui
	Mot	Sortie du registre	%Ri.O	Oui
	Bit	Sortie registre pleine	%Ri.F	Non
	Bit	Sortie registre vide	%Ri.E	Non
%DR	Mot	Numéro du pas courant	%DRi.S	Oui
	Bit	Dernier pas égal au pas courant	%DRi.F	Oui
%FC	Mot	Valeur courante	%FCi.V	Non
	Mot	Valeur de présélection	%FCi.P	Oui
	Bit	Terminé	%FCi.D	Non

Bloc fonction avancé	Mots et bits associés		Adresse	Accès en écriture
%VFC	Mot	Valeur courante	%VFCi.V	Non
	Mot	Valeur de présélection	%VFCi.P	Oui
	Bit	Sens de comptage	%VFCi.U	NON
	Mot	Valeur de capture	%VFCi.C	Non
	Mot	Valeur de seuil 0	%VFCi.SO	Oui
	Mot	Valeur de seuil 0	%VFCi.S1	Oui
	Bit	Débordement	%VFCi.F	Non
	Bit	Fréquence terminée	%VFCi.M	Oui
	Bit	Sortie réflexe 0 activée	%VFCi.R	Oui
	Bit	Sortie réflexe 1 activée	%VFCi.S	Oui
	Bit	Sortie seuil 0	%VFCi.TH0	Non
	Bit	Base temps de la mesure de fréquence	%VFCi.T	Oui
%PWM	Mot	Pourcentage d'impulsions au pas 1 par rapport à la période totale.	%PWMi.R	Oui
	Mot	Période préréglée	%PWMi.P	Oui
%PLS	Mot	Nombre de pulsations	%PLSi.N	Oui
	Mot	Valeur de présélection	%PLSi.P	Oui
	Bit	Sortie courante activée	%PLSi.Q	Non
	Bit	Génération terminée	%PLSi.D	Non
%SBR	Bit	Bit de registre	%SBRi.J	Non
%SC	Bit	Bit de compteur à pas	%SCi.j	Oui
%MSG	Bit	Terminé	%MSGi.D	Non
	Bit	Erreur	%MSGi.E	Non

Principes de programmation de blocs fonctions avancés

Présentation

Les applications Twido sont stockées sous la forme de programmes par listes, et ce, même si ces applications ont été rédigées à l'aide d'un éditeur schéma à contacts. Les automates Twido peuvent ainsi être considérées comme des "machines à listes". Le terme "réversibilité" se rapporte à la capacité de TwidoSoft à convertir une application liste d'instructions en application schémas à contacts, et vice versa. Par défaut, tous les programmes schémas à contacts sont réversibles.
Tout comme les blocs fonctions élémentaires, les blocs fonctions avancés doivent se conformer à des règles de réversibilité. La structure des blocs fonctions réversibles dans le langage liste d'instructions requiert l'utilisation des instructions suivantes :

- **BLK** : marque le début du bloc et la section d'entrée du bloc fonction.
- **OUT_BLK** : marque la fin de la section de sortie du bloc fonction.
- **END_BLK** : marque la fin du bloc fonction.

Note : Il n'est pas nécessaire d'utiliser ces instructions de blocs fonctions réversibles pour un programme par listes d'instructions qui fonctionne correctement. Certaines instructions permettent une programmation en langage liste d'instructions non réversible.

Entrées et sorties dédiées

Les fonctions avancées Compteur rapide (FC), Compteur rapide (VFC), PLS et PWM utilisent des entrées et des sorties dédiées. Ces bits ne sont toutefois pas réservés à une utilisation exclusive par un bloc unique. Au contraire, l'utilisation de ces ressources dédiées doit faire l'objet d'une gestion spécifique.
Lorsque vous utilisez des fonctions avancées, il est nécessaire que vous gériez la méthode d'allocation des entrées et des sorties dédiées. TwidoSoft vous assiste lors de la configuration de ces ressources en affichant des informations de configuration d'E/S et en vous avertissant si une entrée ou une sortie dédiée est déjà utilisée par un bloc fonction configuré (reportez-vous au guide d'exploitation TwidoSoft).
Le tableau suivant résume les dépendances des entrées et des sorties dédiées, ainsi que les fonctions spécifiques.
En cas d'utilisation avec des fonctions de comptage :

Entrées	Utilisation
%I0.0.0	%VFC0 : Gestion Haut/Bas ou Phase B
%I0.0.1	%VFC0 : Entrée d'impulsion ou phase A
%I0.0.2	%FC0 : Entrée d'impulsion ou entrée de présélection %VFC0
%I0.0.3	%FC1 : Entrée d'impulsion ou entrée de capture %VFC0
%I0.0.4	%FC2 : Entrée d'impulsion ou entrée de capture %VFC1
%I0.0.5	Entrée de présélection %VFC1
%I0.0.6	%VFC1 : Gestion Haut/Bas ou Phase B
%I0.0.7	%VFC1 : Entrée d'impulsion ou phase A

En cas d'utilisation avec des fonctions de comptage ou des fonctions spéciales :

Sorties	Utilisation
%Q0.0.0	Sortie %PLS0 ou PWM0
%Q0.0.1	Sortie %PLS1 ou PWM1
%Q0.0.2	Sorties réflexes pour %VFC0
%Q0.0.3	
%Q0.0.4	Sorties réflexes pour %VFC1
%Q0.0.5	

Utilisation d'entrées et de sorties dédiées

TwidoSoft utilise les règles suivantes lors de l'utilisation d'entrées et de sorties dédiées.

- Chaque bloc fonction utilisant des E/S dédiées doit être configuré et référencé dans l'application. L'E/S est uniquement allouée lors de la configuration d'un bloc fonction. Elle ne l'est pas lors de son référencement dans un programme.
- Après qu'un bloc fonction a été configuré, son entrée et sa sortie dédiées ne peuvent pas être utilisées par l'application ou par un autre bloc fonction.
  Par exemple, si vous configurez %PLS0, vous ne pouvez pas utiliser %Q0.0.0 dans %DR0 (programmateur cyclique) ou dans la logique de l'application (ST %Q0.0.0).
- Si une entrée ou une sortie dédiée est requise par un bloc fonction déjà utilisé par l'application ou par un autre bloc fonction, il n'est pas possible de configurer ce bloc fonction.
  Par exemple, si vous configurez %FC0 comme compteur, %VFC0 ne pourra pas être configuré pour utiliser %I0.0.2 comme entrée de capture.

Note : Pour modifier l'utilisation des E/S dédiées, vous devez d'abord supprimer la configuration du bloc fonction en définissant le type d'objet sur "non utilisé", puis supprimer les références au bloc fonction dans votre application.

Bloc fonction registre LIFO/FIFO (%Ri)

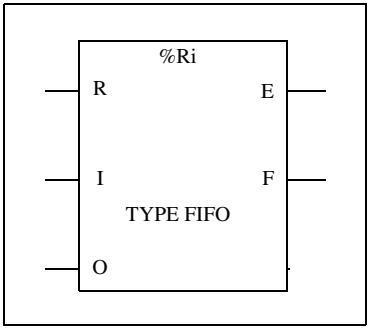
Introduction

Un registre est un bloc mémoire qui permet de stocker jusqu'à 16 mots de 16 bits de deux manières différentes :

- par une file d'attente, appelée « FIFO » (First In, First Out – Premier entré, Premier sorti) ;
- par une pile, appelée « LIFO » (Last In, First Out – Dernier entré, Premier sorti).

Illustration

L'exemple suivant illustre l'utilisation du bloc fonction registre.



Bloc fonction registre

Paramètres

Le bloc fonction registre possède les paramètres suivants :

Paramètre	Etiquette	Valeur
Numéro de registre	%Ri	0 à 3
Type	FIFO LIFO	File d'attente (sélection par défaut) Pile
Mot d'entrée	%Ri.I	Mot d'entrée registre. Peut être lu, testé et écrit.
Mot de sortie	%Ri.O	Mot de sortie registre. Peut être lu, testé et écrit.
Entrée (ou instruction) de stockage	I (In, Entrée)	Sur un front montant, stocke le contenu du mot %Ri.I dans le registre.
Entrée (ou instruction) de récupération	O (Out, Sortie)	Sur un front montant, charge un mot de données dans le mot %Ri.O.
Entrée (ou instruction) RAZ	R (Reset, Remise à zéro)	A l'état 1, initialise le registre.
Sortie « Vide »	E (Empty, Vide)	Le bit %Ri.E associé indique que le registre est vide. Peut être testé.
Sortie « Plein »	F (Full, Plein)	Le bit %Ri.F associé indique que le registre est plein. Peut être testé.

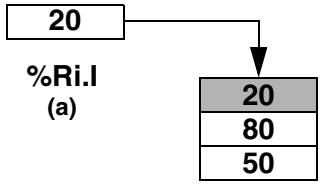
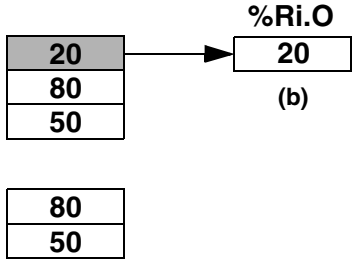
LIFO, fonctionnement

Introduction

En fonctionnement LIFO (Last In, First Out - Dernier entré, Premier sorti), la dernière information entrée est la première à être récupérée.

Fonctionnement

Le tableau suivant décrit le fonctionnement LIFO.

Etape	Description	Exemple
1	A la réception d'une demande de stockage (front montant sur l'entrée I ou activation de l'instruction I), le contenu du mot d'entrée %Ri.I (qui a préalablement été chargé) est stocké au plus haut de la pile (fig. a). Lorsque la pile est pleine (sortie F=1), plus aucun élément ne peut être stocké.	Storage of the contents of %Ri.I at the top of the stack. 
2	A la réception d'une demande de récupération (front montant sur l'entrée O ou activation de l'instruction O), le mot de données le plus haut (le dernier à avoir été entré) est chargé dans le mot %Ri.0 (fig. b). Lorsque le registre est plein (sortie E=1), plus aucun élément ne peut être récupéré.	Retrieval of the data word highest in the stack. 
3	Le mot de sortie %Ri.O n'est pas modifié et sa valeur reste inchangée. La pile peut être réinitialisée à tout moment (état 1 sur l'entrée R ou activation de l'instruction R). L'élément indiqué par le pointeur se retrouve le plus haut dans la pile.	

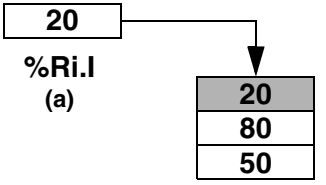
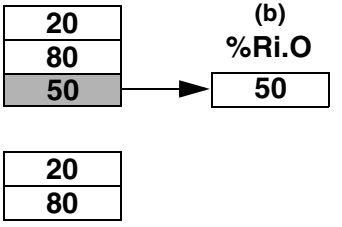
FIFO, fonctionnement

Introduction

En fonctionnement FIFO (First In, First Out - Premier entré, Premier sorti), la première information entrée est la première à être récupérée.

Fonctionnement

Le tableau suivant décrit le fonctionnement FIFO.

Etape	Description	Exemple
1	A la réception d'une demande de stockage (front montant sur l'entrée I ou activation de l'instruction I), le contenu du mot d'entrée %Ri.I (qui a préalablement été chargé) est stocké au plus haut de la file d'attente (fig. a). Lorsque la file d'attente est pleine (sortie F=1), plus aucun élément ne peut être stocké.	Storage of the contents of %Ri.I at the top of the queue. 
2	A la réception d'une demande de récupération (front montant sur l'entrée O ou activation de l'instruction O), le mot de données le moins haut dans la file d'attente est chargé dans le mot de sortie %Ri.O et le contenu du registre est déplacé d'une place vers le bas, dans la file d'attente (fig. b). Lorsque le registre est plein (sortie E=1), plus aucun élément ne peut être récupéré.	Retrieval of the first data item which is then loaded into %Ri.O. 
3	Le mot de sortie %Ri.O n'est pas modifié et sa valeur reste inchangée. La file d'attente peut être réinitialisée à tout moment (état 1 sur l'entrée R ou activation de l'instruction R).	

Programmation et configuration des registres

Introduction

L'exemple de programmation suivant illustre le chargement d'un mot mémoire (%MW34) dans un registre (%R2.I) à la demande de stockage %I0.2, si le registre %R2 n'est pas plein (%R2.F = 0). La demande de stockage dans le registre est faite par %M1. La demande de récupération est faite par l'entrée %I0.3 et %R2.O est chargé dans %MW20, si le registre n'est pas vide (%R2.E = 0).

1. Une demande de stockage dans le registre est faite par %M1.
2. Un mot mémoire (%MW34) est chargé dans un registre (%R2.I). Une demande de stockage à %I0.2 est faite, si le registre %R2 n'est pas plein (%R2.F = 0).
3. Une demande de stockage à %I0.2 est faite, si le registre %R2 n'est pas plein (%R2.F = 0).

Exemple de programmation

L'illustration suivante représente un bloc fonction registre et présente des exemples de programmation réversible et non réversible.

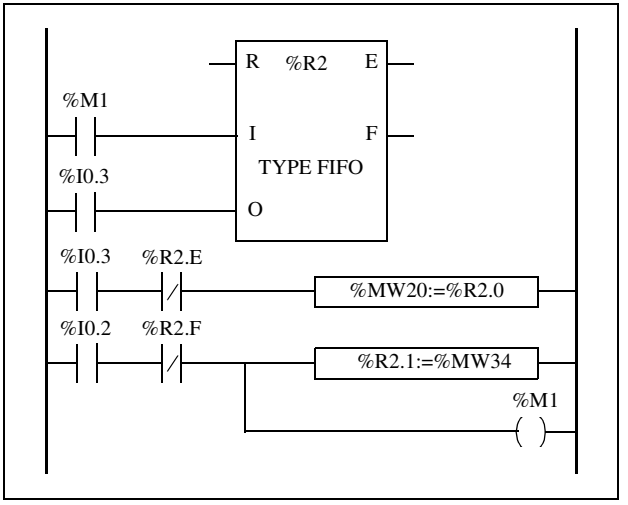


Schéma à contacts

```
BLK          %R2
LD           %M1
I
LD           %I0.3
O
END_BLK
LD           %I0.3
ANDN         %R2.E
[%MW20:=%R2.0]
LD           %I0.2
ANDN         %R2.F
[%R2.1:=%MW34]
ST           %M1
```

Programme réversible

```
LD           %M1
I            %R2
LD           %I0.3
O            %R2
ANDN         %R2.E
[%MW20:=%R2.0]
LD           %I0.2
ANDN         %R2.F
[%R2.1:=%MW34]
ST           %M1
```

Programme non réversible

Configuration

Seul le type du registre devra être entré au cours de la configuration.

- FIFO (par défaut), ou
- LIFO

Cas spéciaux

Le tableau suivant présente une liste des cas spéciaux de programmation et de configuration des registres.

Cas spécial	Description
Effet d'un démarrage à froid (%S0=1)	Provoque l'initialisation du contenu du registre. Le bit de sortie %Ri.E associé à la sortie E est mis à 1.
Effet d'une reprise à chaud (%S1=1) d'un arrêt de l'automate	N'a aucun effet sur la valeur courante du registre ou sur l'état de ses bits de sortie.

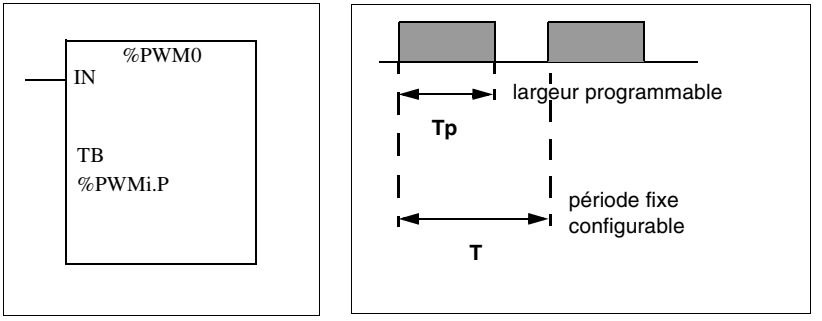
Bloc fonction %PWM (modulation de la largeur d'impulsion)

Introduction

Le bloc fonction de modulation de la largeur d'impulsion (%PWM) génère un signal d'onde carrée sur des voies de sortie dédiées (%Q0.0.0 ou %Q0.0.1). Le bloc fonction %PWM vous permet d'ajuster la largeur du signal, et, par conséquent, le cycle de charge. Les automates disposant de sorties relais pour ces deux voies ne prennent pas en charge cette fonction, en raison d'une limitation de fréquences. Deux blocs %PWM sont disponibles. Le bloc %PWM0 utilise la sortie dédiée %Q0.0.0 et le bloc %PMW1 utilise la sortie dédiée %Q0.0.1. Les blocs fonctions %PLS se partagent les mêmes sorties dédiées. Il est donc nécessaire de choisir l'une ou l'autre des fonctions.

Illustration

Bloc PWM et chronogramme :



Paramètres

Le tableau suivant présente les différents paramètres du bloc fonction PWM.

Paramètre	Etiquette	Description
Base temps	TB	0,1 ms[1], 10 ms, 1 s (valeur par défaut)
Période préréglée	%PWMi.P	0 < %PWMi.P <= 32767 avec une base temps de 10 ms ou 1 s 0 < %PWMi.P <= 255 avec une base temps de 0,57 ms ou 0,142 ms 0 = Fonction non utilisée
Rapport d'impulsion (Cycle de charge)	%PWMi.R	Cette valeur donne le pourcentage du signal à l'état 1 au cours d'une période. Le Tp de largeur est ainsi égal à : Tp = T * (%PWMi.R/100). L'application utilisateur écrit la valeur de %PWMi.R. Ce mot contrôle la modulation de la largeur. Pour plus d'informations sur la définition T, reportez-vous à la section suivante, intitulée "Plage de périodes". La valeur par défaut est 0 et les valeurs supérieures à 100 sont considérées comme étant égales à 100.
Entrée de génération de l'impulsion	IN	A l'état 1, le signal de modulation de la largeur d'impulsion est généré sur la voie de sortie. A l'état 0, la voie de sortie est réglée sur 0.

Note :

1. Nous vous recommandons de ne pas utiliser cette base temps avec des automates Twido disposant de sorties relais.

Plage de périodes

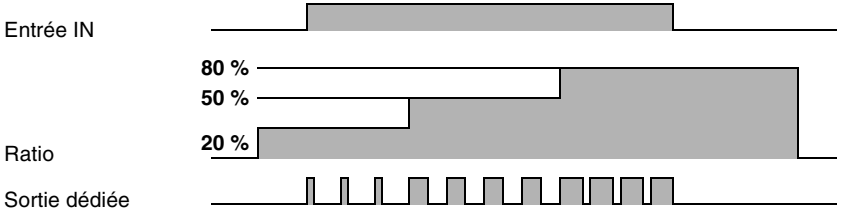
La valeur de présélection et la base temps peuvent être modifiées au moment de la configuration. Ces paramètres sont utilisés pour fixer la période du signal T=%PWMi.P * TB. L'obtention de rapports bas nécessite que le %PWMi.P sélectionné soit d'autant plus élevé. Plage de périodes disponibles :

- 0,142 ms à 36,5 ms en pas de 0,142 ms (27,4 Hz à 7 kHz)
- 0,57 ms à 146 ms en pas de 0,57 ms (6,84 Hz à 1,75 kHz)
- 20 ms à 5,45 min en pas de 10 ms
- 2 s à 9,1 heures en pas de 1 s

Fonctionnement

La fréquence du signal de sortie est réglée au moment de la configuration en sélectionnant la base temps et le %PWMi.P préréglé. La modification du rapport % PWMi.R dans le programme permet de moduler la largeur du signal. L'illustration suivante représente un diagramme d'impulsion du bloc fonction PWM avec différents cycles de charge.

Diagramme d'impulsions du bloc fonction PWM :



Programmation et configuration

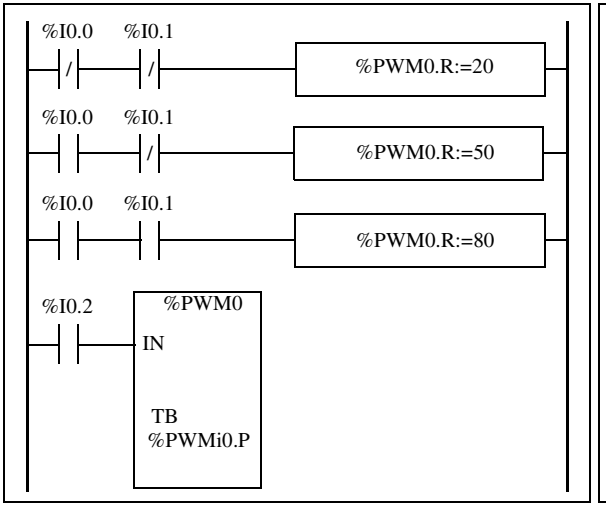
Dans cet exemple, la largeur du signal est modifiée par le programme en fonction de l'état des entrées %I0.0.0 et %I0.0.1 de l'automate.

Si %I0.0.1 et %I0.0.2 sont réglés sur 0, le rapport %PWM0.R est réglé sur 20 % et la durée du signal à l'état 1 est alors égale à : 20 % x 500 ms = 100 ms.

Si %I0.0.0 est réglé sur 0 et %I0.0.1 est réglé sur 1, le rapport %PWM0.R est réglé sur 50 % (durée de 250 ms).

Si %I0.0.0 et %I0.0.1 sont réglés sur 1, le rapport %PWM0.R est réglé sur 80 % (durée de 400 ms).

Exemple de programmation :




```
LDN     %I0.0
ANDN    %I0.1
[%PWM0.R:=20]
LD      %I0.0
ANDN    %I0.1
[%PWM0.R:=50]
LD      %I0.0
AND     %I0.1
[%PWM0.R:=80]
BLK     %PWM0
LD      %I0.2
IN
END_BLK
```

Cas spéciaux

Le tableau suivant présente une liste de cas spéciaux de programmation du bloc fonction PWM.

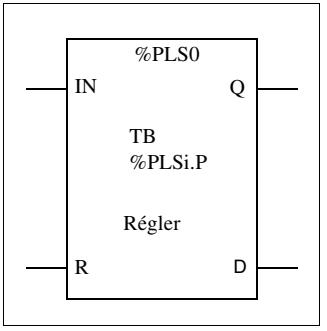
Cas spécial	Description
Effet d'un redémarrage à froid (%S0=1)	Règle le rapport %PWMi.R sur 0. En complément, la valeur de %PWMi.P est rétablie sur sa valeur configurée d'origine et prévaudra sur toute modification apportée dans l'éditeur de tables d'animation ou l'afficheur optionnel.
Effet d'un redémarrage à chaud (%S1=1)	Aucun effet.
Utilisation d'une base temps 0,142 ms ou 0,57 ms	Le fait de forcer la sortie %Q0.0.0 ou %Q0.0.1 à l'aide d'un périphérique de programmation n'interrompt pas la génération du signal.

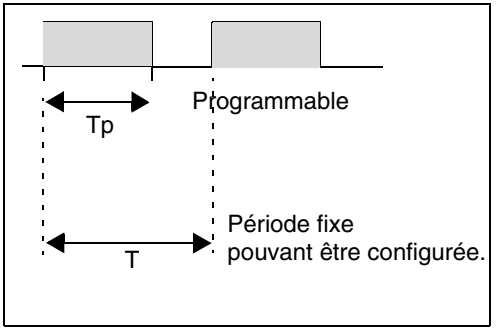
Bloc fonction sortie du générateur d'impulsion (%PLS)

Introduction

Le bloc fonction %PWM (modulation de la largeur d'impulsion) génère un signal d'onde carrée sur une voie de sortie dédiée (%Q0.0.0 ou %Q0.0.1). Le bloc fonction %PWM vous permet d'ajuster la largeur du signal, et, par conséquent, le cycle de charge. Les automates disposant de sorties relais pour ces deux voies ne prennent pas en charge cette fonction, en raison d'une limitation de fréquences.
Deux blocs %PWM sont disponibles. Le bloc %PWM0 utilise la sortie dédiée %Q0.0.0 et le bloc %PMW1 utilise la sortie dédiée %Q0.0.1. Les blocs fonctions %PLS se partagent les mêmes sorties dédiées. Il est donc nécessaire de choisir l'une ou l'autre des fonctions.

Représentation





Caractéristiques Le tableau ci-dessous contient les caractéristiques du bloc fonction PLS.

Fonction	Objet	Description
Base temps	TB	0,142 ms, 0,57 ms, 10 ms, 1 sec
Période préréglée	%PLSi.P	La sortie %PLS1 n'arrête pas l'impulsion lorsque %PLS1.N est atteint. Ceci est valable uniquement pour %PLS0. 0 < %PLSi.P <= 32767 pour une base temps de 10 ms ou 1 sec ; 0 < %PLSi.P <= 255 pour une base temps de 0,57 ms ou 0,142 ms ; 0 = Fonction non utilisée.
Nombre d'impulsions	%PLSi.N	Le nombre d'impulsions à générer sur une période T peut être limité à 0 < %PLSi.N < 32767. La valeur par défaut est réglée sur 0. Pour produire un nombre illimité d'impulsions, réglez %PLSi.N sur zéro. Il est toujours possible de modifier le nombre des impulsions sans tenir compte du paramètre Réglable.
Réglable	Y/N	Lorsqu'il est réglé sur Y, il est possible de modifier la valeur de présélection %PLSi.P via l'afficheur ou l'éditeur de tables d'animation. Lorsqu'il est réglé sur N, il n'est pas possible d'accéder à cette présélection.
Entrée de génération de l'impulsion	IN	A l'état 1, la génération des impulsions se fait sur la voie de sortie dédiée. A l'état 0, la voie de sortie est réglée sur 0.
Entrée RAZ	R	A l'état 1, met à zéro le nombre d'impulsions des sorties %PLSi.Q et %PLSi.D.
Génération d'impulsions sur sortie courante	%PLSi.Q	A l'état 1, il indique que le signal des impulsions est généré sur la voie de sortie dédiée configurée.
Sortie de génération d'impulsion terminée	%PLSi.D	A l'état 1, la génération du signal est terminée. Le nombre voulu d'impulsions a été généré.

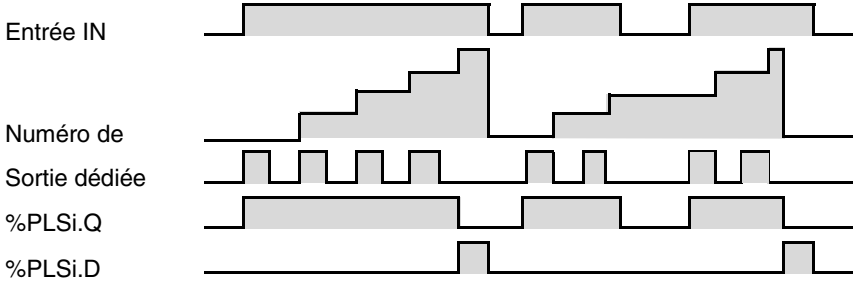
Plage de périodes

La valeur de présélection et la base temps peuvent être modifiées au moment de la configuration. Ces paramètres sont utilisés pour fixer la période du signal T=%PLSi.P * TB. Pour obtenir des rapports bas, le %PLSi.P sélectionné doit être d'autant plus élevé. Plage de périodes disponibles :

- 0,142 ms à 36,5 ms en pas de 0,142 ms (27,4 Hz à 7 kHz)
- 0,57 ms à 146 ms en pas de 0,57 ms (6,84 Hz à 1,75 kHz)
- 20 ms à 5,45 min en pas de 10 ms
- 2 sec à 9,1 heures en pas de 1 sec

Fonctionnement

L'illustration suivante représente le schéma d'impulsion du bloc fonction %PLS.



Cas spéciaux

Cas spécial	Description
Effet d'un démarrage à froid (%S0=1)	Règle la fonction %PLSi.P sur la valeur définie au cours de la configuration.
Effet d'une reprise à chaud (%S1=1)	Aucun effet
Effet d'un arrêt de l'automate	La sortie %Q0.0.0 ou %Q0.0.1 est réglée sur 0 sans prendre en compte l'état du bit système %S8.
Effet de la modification de la valeur de présélection (%PLSi.P)	Prend effet immédiatement
Utilisation d'une base temps de 0,142 ms ou de 0,57 ms	Le fait de forcer la sortie %Q0.0.0 ou %Q0.0.1 à l'aide d'un périphérique de programmation n'interrompt pas la génération du signal.

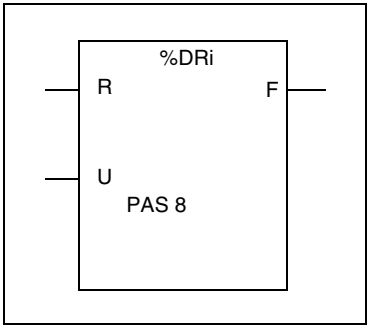
Bloc fonction programmateur cyclique (%DR)

Introduction

Le fonctionnement des programmateurs cycliques est semblable à celui des programmateurs cycliques électromécaniques qui permettent la modification de pas en fonction d'événements externes. A chaque pas, le point haut d'une came donne une commande exécutée par le système de régulation. Dans le cas d'un programmateur cyclique, ces points hauts sont symbolisés par l'état 1 pour chacun des pas et sont affectés aux bits de sortie %Qi.j ou aux bits internes %Mi, appelés "bits de contrôle".

Illustration

L'exemple suivant illustre l'utilisation du bloc fonction programmateur cyclique.



Bloc fonction programmateur cyclique

Paramètres

Le bloc fonction programmateur cyclique possède les paramètres suivants.

Paramètre	Etiquette	Valeur
Numéro	%DRi	Automates compacts 0 à 3 Automates modulaires 0 à 7
Numéro du pas courant	%DRi.S	0-%DRi.S-7. Mot pouvant être lu et écrit. La valeur écrite doit être une valeur décimale immédiate. Une fois écrite, la valeur sera prise en compte à la prochaine exécution du bloc fonction.
Nombre de pas		1 à 8 (par défaut)
Retour à l'entrée (ou à l'instruction) du pas 0	R (Reset, Remise à zéro)	A l'état 1, règle le programmateur cyclique sur le pas 0.
Entrée (ou instruction) avancée	U (Up, Haut)	Sur un front montant, provoque le passage du programmateur cyclique au pas suivant et met à jour les bits de contrôle.
Sortie	F (Full, Plein)	Indique que le pas courant est égal au dernier pas défini. Le bit associé %DRi.F peut être testé (par exemple, %DRi.F=1, si %DRi.S= nombre de pas configurés - 1).
Bits de contrôle		Bits de sortie ou bits internes associés au pas (16 bits de contrôle) et définis dans l'éditeur de configuration.

Fonctionnement des blocs fonctions du programmateur cyclique

Introduction

Le programmateur cyclique est composé des éléments suivants :

- Une matrice de données constantes (des cames), organisée en huit pas (numérotés de 0 à 7) et 16 bits de données (état du pas), disposés en colonnes numérotées de 0 à F.
- Une liste de bits de contrôle (un par colonne) correspondant soit aux sorties %Q0.i ou %Q1.i, soit aux bits internes %Mi. Au cours du pas courant, les bits de contrôle prennent les états binaires définis pour ce pas.

L'exemple présenté dans le tableau suivant résume les caractéristiques principales du programmateur cyclique.

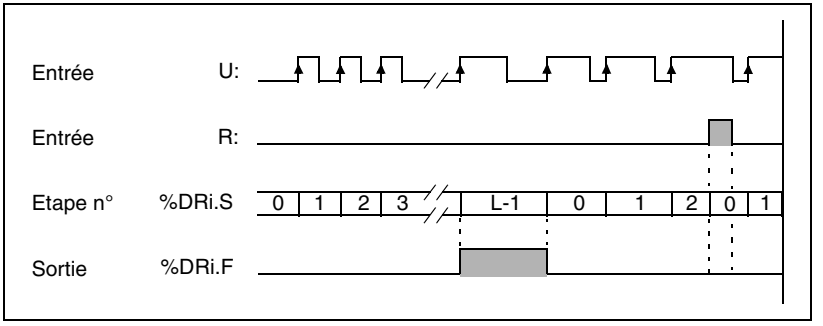
Colonne	**0**	**1**	**2**		**D**	**E**	**F**
Bits de contrôle	%Q0.1	%Q0.3	%Q1.5		%Q0.6	%Q0.5	%Q1.0
Pas 0	0	0	1		1	1	0
Pas 1	1	0	1		1	0	0
Pas 5	1	1	1		0	0	0
Pas 6	0	1	1		0	1	0
Pas 7	1	1	1		1	0	0

Fonctionnement

Dans l'exemple précédent, le pas 5 est le pas courant, les bits de contrôle %Q0.1, %Q0.3 et %Q1.5 sont à l'état 1 ; les bits de contrôle %Q0.6, %Q0.5 et %Q1.0 sont à l'état 0. Le numéro du pas courant est incrémenté d'une unité sur chaque front montant à l'entrée U (ou lors de l'activation de l'instruction U). Le pas courant peut être modifié par le programme.

Chronogramme

Le diagramme suivant illustre la temporisation du fonctionnement du programmateur cyclique.



Cas spéciaux

Le tableau suivant présente une liste des cas spéciaux de fonctionnement du programmateur cyclique.

Cas spécial	Description
Effets d'un redémarrage à froid (%S0=1)	Provoque la réinitialisation du programmateur cyclique au pas 0 (par la mise à jour des bits de contrôle).
Effet d'un redémarrage à chaud (%S1=1)	Met à jour les bits de contrôle après le pas courant.
Effet d'un saut de programme	Si le programmateur cyclique n'est pas scruté, les bits de contrôle ne sont pas remis à 0.
Mise à jour des bits de contrôle	Survient uniquement en cas de changement de pas ou lors d'un démarrage à froid ou d'un redémarrage à chaud.

Programmation et configuration des programmateurs cycliques

Introduction

Dans l'exemple suivant de programmation et de configuration d'un programmateur cyclique, les six premières sorties (%Q0.0 à %Q0.5) sont activées les unes à la suite des autres, chaque fois que l'entrée %I0.1 est réglée sur 1. L'entrée I0.0 remet les sorties à zéro.

Exemple de programmation

L'illustration suivante représente un bloc fonction programmateur cyclique et présente des exemples de programmation réversible et non réversible.

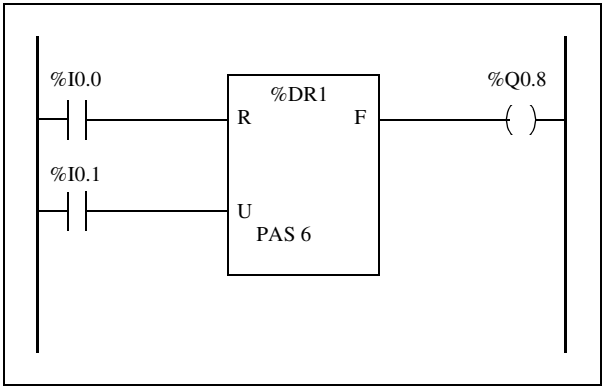


Schéma à contacts

```
BLK     %DR1
LD      %I0.0
R
LD      %I0.1
U
OUT_BLK
LD      F
ST      %Q0.8
END_BLK
```

Programme réversible

```
LD      %I0.0
R       %DR1
LD      %I0.1
U       %DR1
LD      %DR1.F
ST      %Q0.8
```

Programme non réversible

Configuration

Les informations suivantes sont définies au moment de la configuration :

- nombre de pas : 6
- états de sortie (bits de contrôle) pour chaque pas du programmateur cyclique

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Etape 1 :	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Etape 2 :	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Etape 3 :	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
Etape 4 :	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
Etape 5 :	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
Etape 6 :	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0

- affectation des bits de contrôle

1 :	%Q0.0	4 :	%Q0.1
2 :	%Q0.2	5 :	%Q0.3
3 :	%Q0.4	6 :	%Q0.5

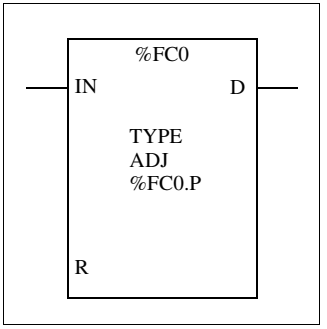
Bloc fonction compteur rapide (%FC)

Introduction

Le bloc fonction compteur rapide (%FC) sert à la fois de compteur et de décompteur. Il peut compter le front montant des entrées TOR pour des fréquences allant jusqu'à 5 kHz. Etant donné que les compteurs rapides (FC) sont gérés par des interruptions matérielles spécifiques, le maintien du taux d'échantillonnage maximal des fréquences peut varier en fonction de la configuration de vos applications et de votre matériel.
La configuration des automates compacts permet l'utilisation d'un maximum de trois compteurs rapides (FC), alors que celle des automates modulaires ne permet que l'utilisation de deux de ces compteurs. Les blocs fonctions compteurs rapides %FC0, %FC1 et %FC2 utilisent respectivement les entrées dédiées %I0.0.2, %I0.0.3 et %I0.0.4. Ces bits ne sont pas réservés exclusivement à ces blocs fonctions. L'affectation de ces bits doit être déterminée selon l'utilisation de ces ressources dédiées par d'autres blocs fonctions.

Illustration

L'exemple suivant illustre un bloc fonction compteur rapide (FC).



Paramètres

Le tableau présente les différents paramètres du bloc fonction compteur rapide (FC).

Paramètre	Etiquette	Description
Direction	TYPE	Paramètre réglé lors de la configuration et permettant de choisir entre le compteur et le décompteur.
Valeur de présélection	%FCi.P	Valeur initiale réglée entre 1 et 65 635.
Réglable	Y/N	Lorsqu'il est réglé sur Y, il est possible de modifier les valeurs de présélection %FCi.P et %FCi.V à l'aide de l'afficheur ou de l'éditeur de tables d'animation. Lorsqu'il est réglé sur N, il n'est pas possible d'accéder à cette présélection.
Valeur courante	%FCi.V	Le comptage des valeurs courantes est effectué de manière croissante ou décroissante selon la fonction de comptage sélectionnée. Pour le comptage, la valeur courante est mise à zéro et peut aller jusqu'à 65 536. Pour le décomptage, la valeur courante est la valeur de présélection %FCi.P et peut décroître jusqu'à zéro.
Activer entrée	IN	A l'état 1, la valeur courante est mise à jour selon les impulsions appliquées à l'entrée physique. A l'état 0, la valeur courante reste inchangée.
Remise à zéro	%FCi.R	Paramètre utilisé pour initialiser le bloc. A l'état 1, la valeur courante est mise à 0 lorsque le bloc est configuré en tant que compteur et à %FCi.P lorsqu'il est configuré en tant que décompteur. Le bit Terminé %FCi.D revient à sa valeur par défaut.
Terminé	%FCi.D	Ce bit est réglé sur 1 lorsque %FCi.V atteint %FCi.P si celui-ci est configuré en tant que compteur, ou lorsque %FCi.V atteint zéro s'il est configuré en tant que décompteur. Ce bit en lecture seule est mis à 0 uniquement lorsque le paramètre %FCi.R est réglé sur 1.

Remarque

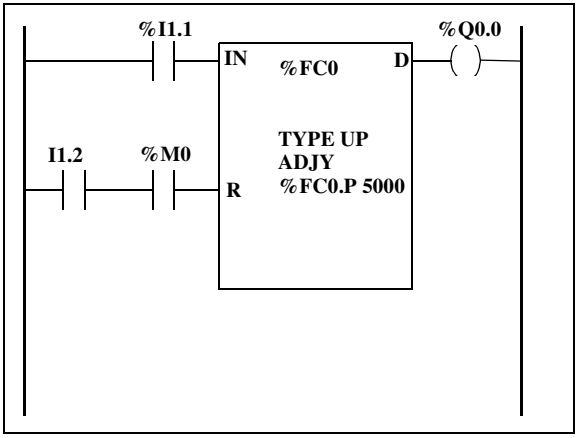
Lorsque le bloc est configuré comme réglable, l'application peut modifier la valeur de présélection %FCi.P et la valeur courante %FCi.V à tout moment. Cependant, une nouvelle valeur est prise en compte uniquement lorsque la réinitialisation des entrées est active ou sur le front montant de la sortie %FCi.D. Cela permet d'effectuer plusieurs comptages successifs sans perdre une seule impulsion.

Fonctionnement

Lorsque le bloc est configuré pour le comptage, la valeur courante est incrémentée de 1 dès qu'un front montant apparaît à l'entrée dédiée. Lorsque la valeur est égale à la valeur de présélection %FCi.P, le bit de sortie Terminé %FCi.D est réglé sur 1 et la valeur courante %FCi.V devient égale à zéro.
Lorsque le bloc est configuré pour le décomptage, la valeur courante est diminuée de 1 dès qu'un front montant apparaît à l'entrée dédiée. Lorsque la valeur est égale à zéro, le bit de sortie Terminé %FCi.D est réglé sur 1 et la valeur courante devient égale à la valeur de présélection %FCi.P.

Configuration et programmation

Dans l'exemple ci-dessous, l'application compte le nombre des éléments (5000 maximum) et %I1.1 est réglé sur 1. L'entrée pour %FC0 est l'entrée dédiée %I0.0.2. Lorsque la valeur de présélection est atteinte, %FC0.D est réglé et conserve la même valeur jusqu'à ce que %FC0.R soit défini par le résultat de l'opération booléenne « ET » sur %I1.2 et %M0.




```
BLK      %FC0
LD       %I1.1
IN
LD       %I1.2
AND      %MO
R
OUT_BLK
LD D
ST %Q0.0
END_BLK
```

Cas spéciaux

Le tableau suivant présente une liste des cas spéciaux de programmation du bloc fonction %FC.

Cas spécial	Description
Effet d'un démarrage à froid (%S0=1)	Utilise les valeurs configurées par l'utilisateur ou par l'application utilisateur pour régler tous les attributs %FC.
Effet d'une reprise à chaud (%S1=1)	Aucun effet.
Effet de l'arrêt de l'automate	%FC continue à compter selon les réglages des attributs effectifs au moment de l'arrêt de l'automate.

Bloc fonction compteur rapide (%VFC)

Introduction

Le bloc fonction compteur rapide (%VFC) peut être configuré à l'aide de TwidoSoft pour effectuer l'une des fonctions suivantes :

- compteur/décompteur
- compteur/décompteur bi-phases
- compteur simple
- décompteur simple
- fréquencemètre

Le %VFC permet de compter les entrées TOR pour des fréquences allant jusqu'à 20 kHz. Il est possible de configurer un compteur rapide (VFC) à l'aide des automates compacts et jusqu'à deux compteurs rapides (VFC) à l'aide des automates modulaires.

Affectations E/S dédiées

Les blocs fonctions compteurs rapides (VFC) utilisent des entrées dédiées et des entrées et sorties auxiliaires. Ces entrées et ces sorties ne sont pas réservées exclusivement à ces blocs fonctions. Leur affectation doit être décidée en prenant en compte l'utilisation par d'autres blocs fonctions de ces ressources dédiées. Le tableau suivant récapitule ces affectations.

		Entrées principales		**Entrées auxiliaires**		**Sorties réflexes**	
%VFC	**Utilisation choisie**	**Première entrée (impulsions) IA**	**Seconde entrée (impulsions ou UP/DO) IB**	**Entrée de présélection IPres**	**Entrée d'interception Ica**	**Première sortie réflexe**	**Seconde sortie réflexe**
	Compteur/ décompteur	%I0.0.1 (Impulsions)	%I0.0.0 (Indique UP=1/DO=0)	%I0.0.2 Facultatif	%I0.0.3 Facultatif	%Q0.0.2 Facultatif	%Q0.0.3 Facultatif
	Compteur/ Décompteur bi-phases	%I0.0.1 (Impulsions)	%I0.0.0 (Impulsion phase B)	%I0.0.2 Facultatif	%I0.0.3 Facultatif	%Q0.0.2 Facultatif	%Q0.0.3 Facultatif
	Compteur simple	%I0.0.1 (Impulsions)	Non utilisé	%I0.0.2 Facultatif	%I0.0.3 Facultatif	%Q0.0.2 Facultatif	%Q0.0.3 Facultatif
	Décompteur simple	%I0.0.1 (Impulsions)	Non utilisé	%I0.0.2 Facultatif	%I0.0.3 Facultatif	%Q0.0.2 Facultatif	%Q0.0.3 Facultatif
	Compteur de fréquence	%I0.0.1 (Impulsions)	Non utilisé	Non utilisé	Non utilisé	Non utilisé	Non utilisé
	Compteur/ décompteur	%I0.0.7 (Impulsions)	%I0.0.6 (Indique UP=1/DO=0)	%I0.0.5 Facultatif	%I0.0.4 Facultatif	%Q0.0.4 Facultatif	%Q0.0.5 Facultatif
	Compteur/ Décompteur bi-phases	%I0.0.7 (Impulsions)	%I0.0.6 (Impulsion phase B)	%I0.0.5 Facultatif	%I0.0.4 Facultatif	%Q0.0.4 Facultatif	%Q0.0.5 Facultatif
	Compteur simple	%I0.0.7 (Impulsions)	Non utilisé	%I0.0.5 Facultatif	%I0.0.4 Facultatif	%Q0.0.4 Facultatif	%Q0.0.5 Facultatif
	Décompteur simple	%I0.0.7 (Impulsions)	Non utilisé	%I0.0.5 Facultatif	%I0.0.4 Facultatif	%Q0.0.4 Facultatif	%Q0.0.5 Facultatif
	Fréquencemètre	%I0.0.7 (Impulsions)	Non utilisé	Non utilisé	Non utilisé	Non utilisé	Non utilisé

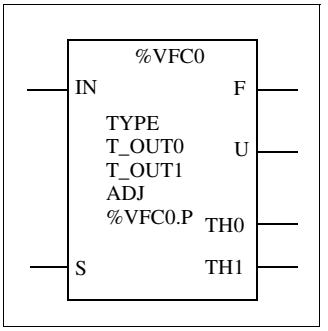
Commentaires :

HA/BA = HAUT/BAS
Util. fac. = utilisation facultative

Lorsqu'elle n'est pas utilisée, l'entrée ou la sortie reste une E/S TOR normale gérée par l'application au cours du cycle principal.

Si %I0.0.2 est utilisé, %FC0 n'est pas disponible.
Si %I0.0.3 est utilisé, %FC2 n'est pas disponible.
Si %I0.0.4 est utilisé, %FC3 n'est pas disponible.

Illustration La figure suivante représente le compteur rapide (VFC) sous forme de bloc.



Paramètres Le tableau suivant répertorie les caractéristiques du bloc fonction compteur rapide (VFC).

Fonction	Description	Valeurs	Utilisation du VFC[4]	Accès en cours d'exécution
Valeur courante (%VFCi.V)	La valeur courante est augmentée ou diminuée en fonction des entrées physiques et de la fonction sélectionnées. Vous pouvez régler cette valeur ou lui réattribuer sa valeur initiale à l'aide de la fonction Régler entrée (%VFCi.S).	0 -> 65 535	CM	Lecture
Valeur de présélection (%VFCi.P)	Uniquement utilisée par la fonction compteur/ décompteur et par le comptage ou le décomptage simple.	0 -> 65 535	CM ou FM	Lecture et écriture[1]
Valeur de capture	Uniquement utilisée par la fonction compteur/ décompteur et par le comptage ou le décomptage simple.	0 -> 65 535	CM	Lecture
Sens de comptage (%VFCi.U)	Défini par le système, ce bit est utilisé par la fonction de comptage/décomptage pour vous indiquer le sens de comptage. Lorsqu'il est réglé sur 1, le comptage est croissant et lorsqu'il est réglé sur 0, le comptage est décroissant. %I0.0.0 détermine, en tant que compteur/décompteur mono-phase, le sens de %VFC0, et %I0.0.6 le détermine pour %VFC1. Pour un compteur/décompteur bi-phases, la différence de phase entre les deux signaux détermine ce sens de comptage. Pour %VFC0, %I0.0 est dédié à IB et %I0.1 à IA. Pour %VFC1, %I0.6 est dédié à IB et %I0.7 à IA.	0 (Décroissant) 1 (Croissant)	CM	Lecture

Fonction	Description	Valeurs	Utilisation du VFC[4]	Accès en cours d'exécution
Activer sortie réflexe 0 (%VFCi.R)	Activer sortie réflexe 0	0 (Désactivé) 1 (Activé)	CM	Lecture et écriture[2]
Activer sortie réflexe 1 (%VFCi.S)	Activer sortie réflexe 1	0 (Désactivé) 1 (Activé)	CM	Lecture et écriture[2]
Valeur de seuil S0 (%VFCi.S0)	Contient la valeur de seuil 0. Sa signification est définie lors de la configuration du bloc fonction. Note : Cette valeur doit être inférieure à %VFCi.S1.	0 -> 65 535	CM	Lecture et écriture[2]
Valeur de seuil S1 (%VFCi.S1)	Contient la valeur de seuil 0. Sa signification est définie lors de la configuration du bloc fonction. Note : Cette valeur doit être supérieure à %VFCi.S0.	0 -> 65 535	CM	Lecture et écriture[1]
Mesure de fréquence valide (%VFCi.M)	Bit utilisé afin de déterminer si l'automate a terminé une mesure de fréquence.	0 (Non valide) 1 (Valide)	FM	Lecture et écriture
Base temps de la mesure de fréquence (%VFCi.T)	Elément de configuration de la base temps (100 ou 1000 millisecondes).	1000 ou 100	FM	Lecture et écriture[1]
Réglable (Y/N)	Elément de configuration qui, lorsqu'il est sélectionné, permet à l'utilisateur de modifier les valeurs de présélection, de seuil et de base temps de la fréquence de mesure en cours d'exécution.	0 (Non) 1 (Oui)	CM ou FM	Non
Activer entrée (IN)	Utilisée pour valider ou inhiber la fonction courante.	0 (Non)	CM ou FM	Lecture et écriture[3]
Régler entrée (S)	Dépend de la configuration à l'état 1 : ● comptage/décomptage ou décomptage : règle la valeur courante sur la valeur de présélection. ● comptage simple : règle la valeur courante sur 0. Cette fonction permet d'initialiser également la commande des sorties seuils et prend en compte toutes les modifications apportées par un utilisateur aux valeurs de seuils définies par l'afficheur ou le programme utilisateur.	0 ou 1	CM ou FM	Lecture et écriture
Sortie pour débordement (F)	Définie sur 1 si %VFCi.V passe de 0 à 65 535 ou de 65 535 à 0. Cette valeur est effacée lorsque la valeur de présélection est définie à l'aide d'une entrée TOR ou d'une instruction S, ou lors d'un redémarrage à froid.	0 ou 1	CM	Lecture

Fonction	Description	Valeurs	Utilisation du VFC[4]	Accès en cours d'exécution
Seuil Bit 0 (%VFCi.TH0)	Défini sur 1 lorsque la valeur courante est supérieure ou égale à la valeur de seuil %VFCi.S0. Nous conseillons de tester ce bit une seule fois dans le programme, car il est mis à jour en temps réel. L'application utilisateur est responsable de la validité de la valeur au moment de son utilisation.	0 ou 1	CM	Lecture
Seuil Bit 1 (%VFCi.TH1)	Défini sur 1 lorsque la valeur courante est supérieure ou égale à la valeur de seuil %VFCi.S1. Nous conseillons de tester ce bit une seule fois dans le programme, car il est mis à jour en temps réel. L'application utilisateur est responsable de la validité de la valeur au moment de son utilisation.	0 ou 1	CM	Lecture

Note :
1. Accessible en écriture uniquement si la fonction Réglable est réglée sur un.
2. Accès disponible si configuré uniquement.
3. Accès en lecture et en écriture seulement à partir de l'application. Accès impossible à partir de l'Afficheur ou de l'Editeur de tables d'animation.
4. CM = Mode comptage et FM = Mode Fréquencemètre.

Description de la fonction de comptage

La fonction de comptage rapide (VFC) fonctionne à une fréquence maximale de 20 kHz et pour une plage de valeurs allant de 0 à 65 535. Les impulsions à compter sont appliquées de la manière suivante.

Fonction	Description	%VFC0 IA ... IB		IA ... IB IA ... IB	
Compteur/ Décompteur	Les impulsions sont appliquées à l'entrée physique ; l'opération courante (augmentation/diminution) est définie par l'état de l'entrée physique IB.	%I0.0.1	%I0.0.0	%I0.0.7	%I0.0.6
Compteur/ Décompteur bi-phases	Les deux phases du codeur sont appliquées aux entrées physiques IA et IB.	%I0.0.1	%I0.0.0	%I0.0.7	%I0.0.6
Compteur simple	Les impulsions sont appliquées à l'entrée physique IA. IB n'est pas utilisée.	%I0.0.1	ND	%I0.0.7	ND
Décompteur simple	Les impulsions sont appliquées à l'entrée physique IA. IB n'est pas utilisée.	%I0.0.1	ND	%I0.0.7	ND

Remarques sur les blocs fonctions

Les opérations d'augmentation ou de diminution sont effectuées sur le front montant des impulsions et ce, uniquement lorsque la fonction comptage est activée.
Deux entrées facultatives sont utilisées en mode comptage. ICa et IPres. ICa est utilisée pour capturer la valeur courante (%VFCi.V) et la stocker dans %VFCi.C. Les entrées Ica sont définies en tant que %I0.0.3 pour %VFC0 et %I0.0.4 pour %VFC1, si cette valeur est disponible.
Lorsque l'entrée IPres est active, la valeur courante est affectée de la manière suivante :

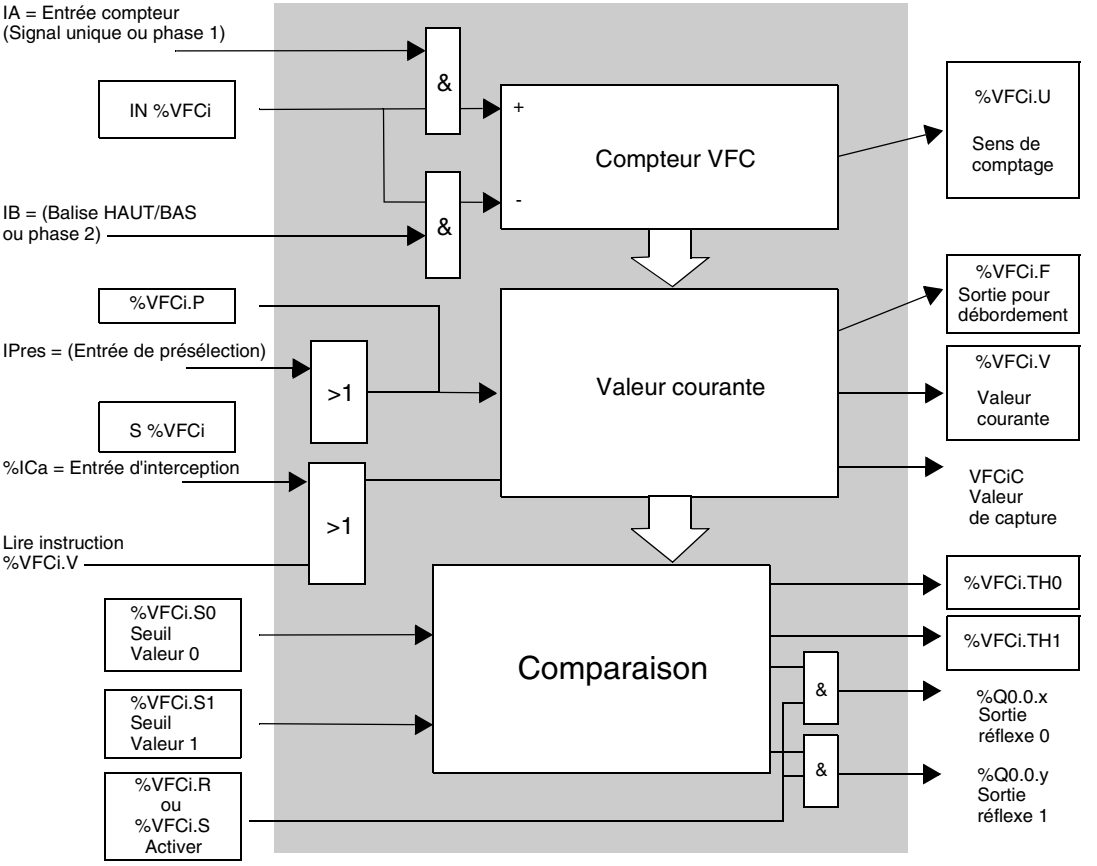
- Pour le comptage, %VFCi.V est remis à 0.
- Pour le décomptage, %VFCi.V est réglé sur %VFCi.P.
- Pour le comptage de fréquence, %VFCi.V et VFCi.M sont réglés sur 0.

Attention : %VFCi.F sera également réglé sur 0. Les entrées IPres sont définies sur %I0.0.2 pour %VFC0 et sur %I0.0.5 pour %VFC1 si cette valeur est disponible.

Remarques sur les sorties blocs fonctions

Pour toutes les fonctions, les valeurs courantes sont comparées aux deux seuils (%VFCi.S0 et % VFCi.S1). Les deux objets bits (%VFCi.TH0 et %VFCi.TH1) sont réglés en fonction des résultats de cette comparaison. Ils sont réglés sur 1 lorsque la valeur courante est supérieure ou égale au seuil correspondant ou sur 0 dans le cas contraire. Les sorties réflexes (si elles sont configurées) sont réglées en fonction de ces comparaisons. Note : Aucune, une ou deux sorties peuvent être configurées. %VFC.U est une sortie du FB. Elle indique le sens de variation du compteur (1 pour comptage, 0 pour décomptage).

Schéma de la fonction de comptage

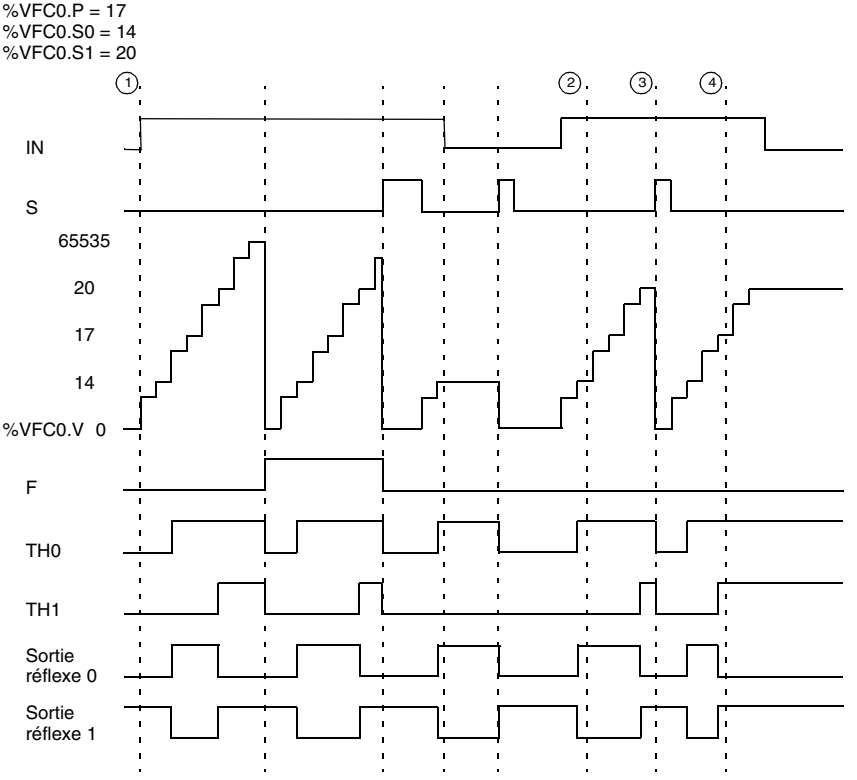


Opération de comptage simple

Voici un exemple de l'utilisation de %VFC en mode comptage simple. Les éléments de configuration suivants ont été définis pour cet exemple :
La valeur de présélection, %VFC0.P, est égale à 17. Le seuil inférieur, %VFC0.S0, est égal à 14 et le seuil supérieur, %VFC0.S1, à 20.

Sortie réflexe	<%VFC.S0	%VFC0.S0 <= < %VFC0.S1	>= %VFC0.S1
%Q0.0.2		X	
%Q0.0.3	X		X

Exemple de chronogramme :



① : %VFC0.U = 1 car VFC est un compteur

② : modification de %VFC0.S1 sur 17

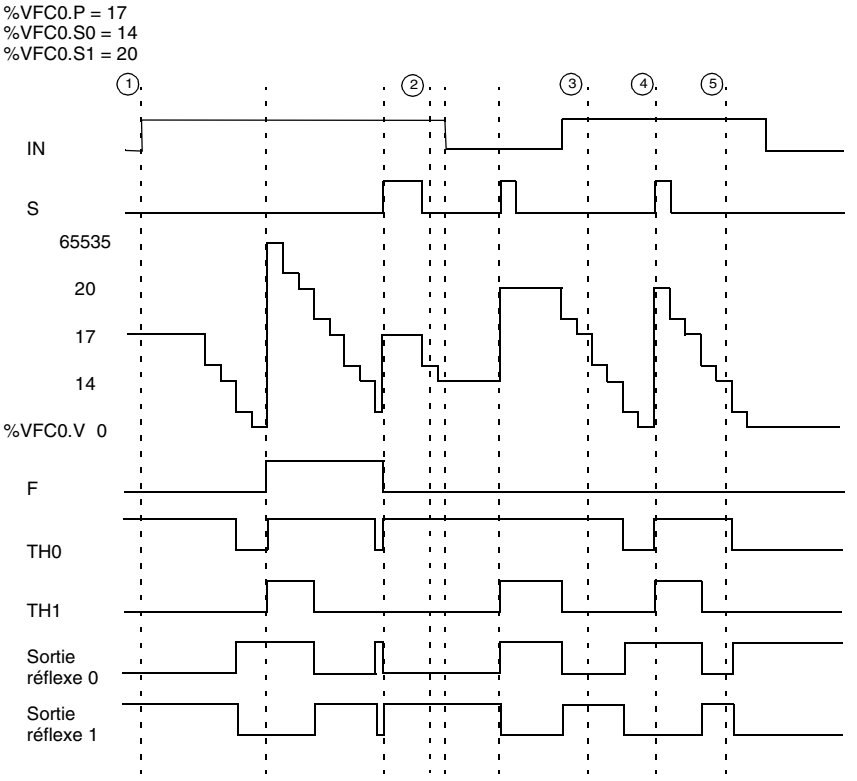
③ : L'activation de l'entrée S permet d'accorder la nouvelle valeur du seuil S1 lors du décompte suivant

④ : une interception de la valeur courante a lieu, ainsi, %VFC0.C = 17

Opération de décomptage simple

Voici un exemple de l'utilisation de %VFC en mode décomptage simple. Les éléments de configuration suivants ont été définis pour cet exemple :
La valeur de présélection, %VFC0.P, est égale à 17. Le seuil inférieur, %VFC0.S0, est égal à 14 et le seuil supérieur, %VFC0.S1, à 20.

Sortie réflexe	<%VFC.S0	%VFC0.S0 <= < %VFC0.S1	>= %VFC0.S1
%Q0.0.2		X	
%Q0.0.3	X		X



① : %VFC0.U = 1 car VFC est un décompteur

② : modification de %VFC0.P sur 20

③ : modification de %VFC0.S1 sur 17

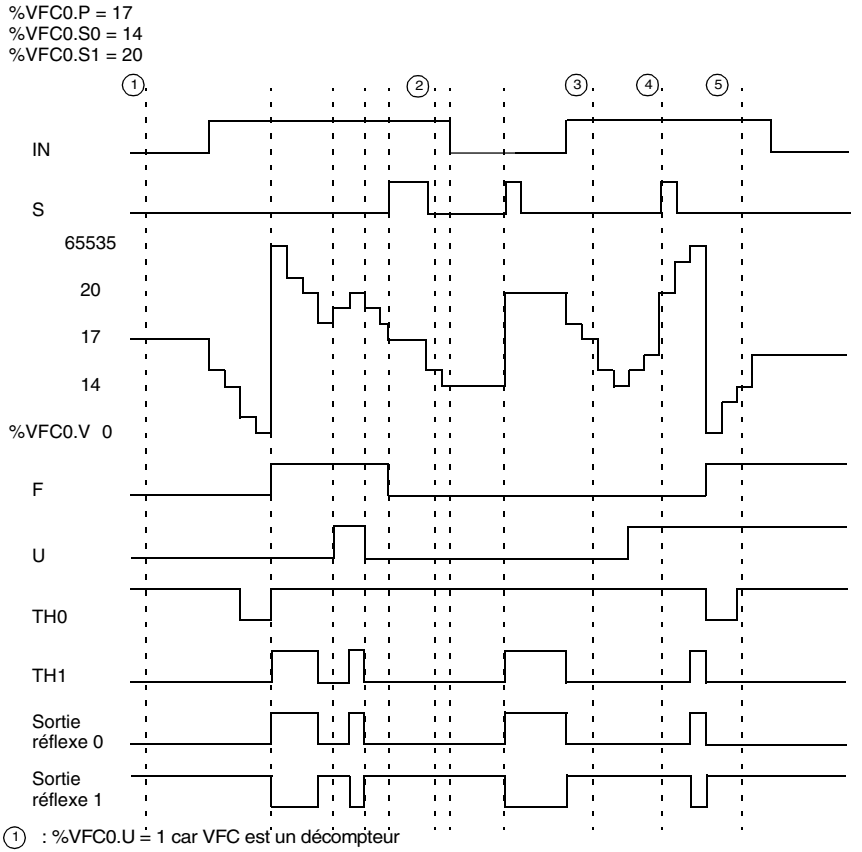
④ : L'activation de l'entrée S permet d'accorder la nouvelle valeur du seuil S1 lors du décompte suivant

⑤ : une interception de la valeur courante a lieu, ainsi, %VFC0.C = 17

Opération de comptage/ décomptage

Voici un exemple de l'utilisation de %VFC en mode comptage/décomptage. Les éléments de configuration suivants ont été définis pour cet exemple :
La valeur de présélection, %VFC0.P, est égale à 17. Le seuil inférieur, %VFC0.S0, est égal à 14 et le seuil supérieur, %VFC0.S1, à 20.

Sortie réflexe	<%VFC.S0	%VFC0.S0 <= < %VFC0.S1	%VFC0.S1
%Q0.0.2		X	
%Q0.0.3	X		X



(1) : %VFC0.U = 1 car VFC est un décompteur

(2) : modification de %VFC0.P sur 20

(3) : modification de %VFC0.S1 sur 17

(4) : L'activation de l'entrée S permet d'accorder la nouvelle valeur du seuil S1 lors du décompte suivant

(5) : une interception de la valeur courante a lieu, ainsi, %VFC0.C = 17

Description de la fonction Fréquencemètre

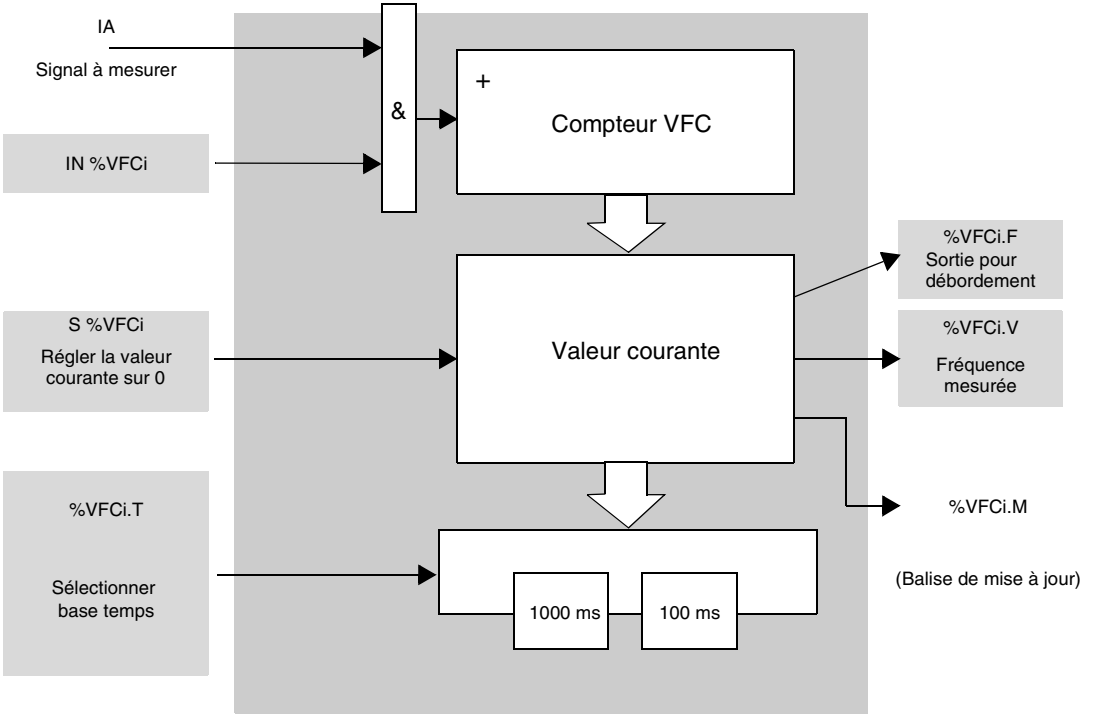
La fonction Fréquencemètre d'un %VFC est utilisée pour mesurer la fréquence en Hz d'un signal périodique sur l'entrée IA. La plage de fréquences pouvant être mesurées s'étend de 10 à 20 Hz. L'utilisateur peut choisir entre deux bases temps. Ce choix est effectué par le biais d'un nouvel objet %VFC.T (Base temps). Une valeur de 100 correspond à une base temps de 100 ms et une valeur de 1000 correspond à une base temps d'une seconde.

Base Temps	Plage de mesure	Précision	Mise à jour
100 ms	100 Hz à 20 Khz	0,05 % pour 20 kHz 10 % pour 100 Hz	10 fois par seconde
1 s	10 Hz à 20 Khz	0,005% pour 20 kHz 10 % pour 10 Hz	Une fois par seconde

L'objet %VFC.M (Mesure de fréquence valide) est réglé sur 1 afin d'indiquer que la mesure est terminée.

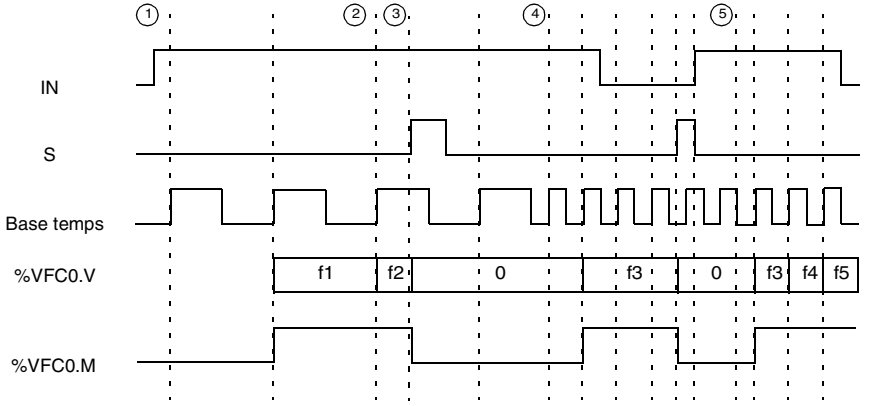
Schéma de la fonction Fréquencemètre

Exemple de schéma de fonction Fréquencemètre :



Opération Fréquencemètre

Voici un exemple de chronogramme de l'utilisation de %VFC en mode Fréquencemètre :



(1) : La mesure de la première fréquence débute ici.

(2) : La valeur de la fréquence courante est mise à jour.

(3) : L'activation de l'entrée S règle %VFC0.V sur 0.

(4) : Modification de %VFC0.T sur 100 ms : cette modification annule la mesure courante et en débute une autre.

(5) : %VFC0.M est réglé sur 0 par l'utilisateur.

Cas spéciaux

Le tableau suivant présente une liste des cas spéciaux de programmation du bloc fonction %VFC.

Cas spécial	Description
Effet d'un redémarrage à froid (%S0=1)	Utilise les valeurs configurées par l'utilisateur ou par l'application utilisateur pour régler tous les attributs %VFC.
Effet d'un redémarrage à chaud (%S1=1)	Aucun effet
Effet de l'arrêt de l'automate	Le %VFC arrête de fonctionner et les sorties restent dans leur état courant.

Transmission et réception de messages - Instruction d'échange (EXCH)

Introduction

Il est possible de configurer un automate Twido afin qu'il puisse communiquer avec des périphériques esclaves Modbus ou envoyer et recevoir des messages en mode ASCII (mode caractères).
TwidoSoft propose les fonctions suivantes pour ces communications :

- Instruction EXCH pour la transmission ou la réception de messages
- Bloc fonction de contrôle d'échange %MSG assurant le contrôle des échanges de données

L'automate Twido utilise le protocole configuré pour le port spécifié lors du traitement d'une instruction EXCH. Chaque port de communication peut être configuré pour un ou plusieurs protocoles. De plus, l'accès à l'instruction EXCH ou au bloc fonction %MSG de chaque port de communication s'effectue par l'ajout du numéro du port (1 ou 2).

EXCH, instruction

L'instruction EXCH permet à un automate Twido d'envoyer ou de recevoir des informations vers/depuis des périphériques ASCII. L'utilisateur définit une table de mots (%MWi:L ou %KWi:L) contenant les données à envoyer ou à recevoir (gestion d'un maximum de 64 mots de données en transmission ou en réception). Le format des tables de mots fait l'objet d'une description détaillée dans les sections relatives à chaque protocole. Un échange de message est exécuté à l'aide de l'instruction EXCH.

Syntaxe

Vous trouverez ci-dessous la syntaxe à utiliser pour l'instruction EXCH :
[EXCHx %MWi:L] ou [EXCHx %KWi:L
Où : x = numéro de port (1 ou 2); L = nombre de mots de la table de mots. Les valeurs contenues dans la table de mots interne %MWi:L prennent la forme i+L - 255.]
L'automate Twido doit terminer l'échange ordonné par la première instruction EXCHx avant qu'une nouvelle instruction d'échange puisse être lancée. Le bloc fonction %MSG doit être utilisé lors de l'envoi de plusieurs messages.

Bloc fonction de contrôle d'échange (%MSG)

Introduction

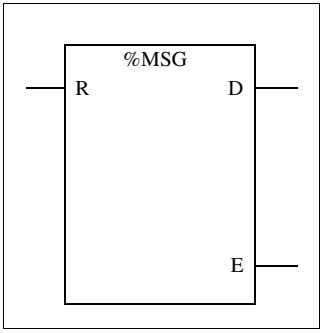
Le bloc fonction %MSG assure la gestion des échanges de données. Ce bloc a trois fonctions :

- vérification des erreurs de communications :
  Cette fonction a pour but de s'assurer que la longueur du bloc (table de mots) programmée avec l'instruction EXCH est suffisante pour le stockage du message à envoyer (comparaison de la longueur programmée dans l'octet le moins significatif du premier mot de la table de mots).
- coordination de plusieurs messages :
  Afin d'assurer la coordination de l'envoi de plusieurs messages, le bloc fonction %MSG contient des informations permettant de s'assurer que la transmission de chaque message est bien terminée.
- transmission de messages prioritaires :
  Le bloc fonction %MSG vous permet de suspendre la transmission d'un message afin d'envoyer un message plus urgent.

La programmation du bloc fonction %MSG est facultative.

Illustration

L'exemple suivant illustre le bloc fonction %MSG.



Paramètres

Le tableau présente les différents paramètres du bloc fonction %MSG.

Paramètre	Etiquette	Valeur
Entrée (ou instruction) RAZ	R	A l'état 1, réinitialise la communication : %MSG.E = 0 et %MSG.D = 1.
Sortie comm. terminée	%MSG.D	Etat 1, comm. terminée, si : • fin de transmission (si transmission) • fin de réception (réception du caractère de fin) • erreur • réinitialise le bloc Etat 0, requête en cours.
Sortie Défaillance (Erreur)	%MSG.E	Etat 1, comm. terminée, si : • commande incorrecte • table configurée de manière incorrecte • réception d'un caractère incorrect (vitesse, parité, etc.) • table de réception pleine (non mise à jour) Etat 0, longueur du message OK, lien OK.

Si une erreur survient lors de l'exécution d'une instruction EXCH, les bits %MSG.D et %MSG.E sont réglés sur 1, le mot système %SW63 contient le code de l'erreur du Port 1 et le mot système %SW64 celui du Port 2. Reportez-vous à la rubrique *Mots système (%SW), p. 341*.

Entrée RAZ (R)

Lorsque Entrée RAZ est réglée sur 1 :
- La transmission de tous les messages est interrompue.
- La sortie Défaillance (Erreur) est remise à 0.
- Le bit Terminé est réglé sur 1.

Un nouveau message peut être envoyé.

Sortie Défaillance (Erreur) (%MSG.E)

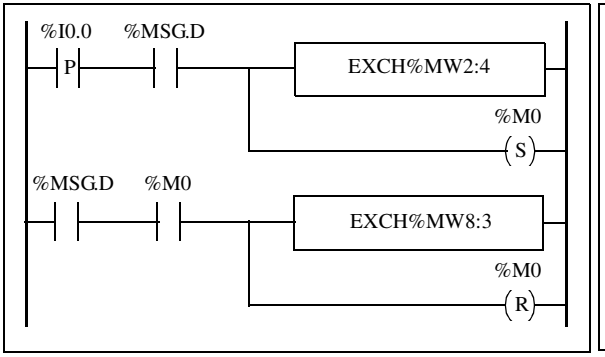
La sortie Erreur est réglée sur 1 en cas d'erreur de programmation des communications ou de transmission d'un message. La sortie Erreur est réglée sur 1 si le nombre d'octets définis dans le bloc de données associé à l'instruction EXCH (mot 1, octet le moins significatif) est supérieur à 128 (80 en hexadécimal).
La sortie Erreur est également réglée sur 1 en cas de problème lors de l'envoi d'un message Modbus vers un périphérique Modbus. Dans ce cas, l'utilisateur devra vérifier la connexion et s'assurer que le périphérique de destination peut recevoir des communications Modbus.

Sortie Communication terminée (%MSG.D)

Lorsque la sortie Terminé est réglée sur 1, l'automate Twido est prêt à envoyer un autre message. L'utilisation de la sortie %MSG.D est recommandée en cas d'envoi de plusieurs messages. Si cette sortie n'est pas utilisée, les messages pourront être perdus.

Transmission de plusieurs messages successifs

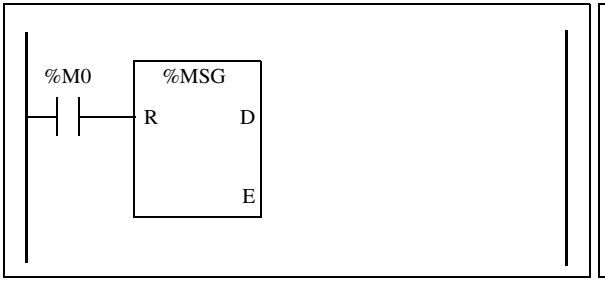
L'exécution de l'instruction EXCH permet d'activer un bloc message dans le programme d'application. Le message est transmis si le bloc message n'est pas déjà actif (%MSG.D = 1). Si plusieurs messages sont envoyés au cours du même cycle, seul le premier message est transmis. La gestion de la transmission de plusieurs messages à l'aide du programme incombe à l'utilisateur.
Exemple de transmission de deux messages successifs :




```
LDR     %I0.0
AND     %MSG.D
[EXCH %MW2:4]
S       %M0
LD      %MSG.D
AND     %M0
[EXCH %MW8:3]
R       %M0
```

Réinitialisation des échanges

L'annulation d'un échange survient lors de l'activation de l'entrée (ou de l'instruction) R. Cette entrée initialise la communication, remet à zéro la sortie %MSG.E et règle la sortie %MSG.D sur 1. Notez qu'il est possible de réinitialiser une communication si une défaillance est détectée.
Exemple de réinitialisation d'un échange :




```
BLK     %MSG
LD      %M0
R
END_BLK
```

Cas spéciaux

Le tableau présente les cas spéciaux de programmation du bloc fonction %MSG.

Cas spécial	Description
Effet d'un démarrage à froid (%S0=1)	Force la réinitialisation de la communication.
Effet d'une reprise à chaud (%S1=1)	Aucun effet.
Effet d'un arrêt de l'automate	Si un message est en cours de transmission, l'automate interrompt le transfert et réinitialise les sorties %MSG.D et %MSG.E.

13.2 Fonctions horodateur

En bref...

Présentation

Cette rubrique offre une description des fonctions de gestion du temps des automates Twido.

Contenu de ce sous-chapitre

Ce sous-chapitre contient les sujets suivants :

Fonctions horloges

Introduction

Les automates Twido possèdent une fonction Date/Heure. Cette fonction requiert l'option Horodateur (RTC) et permet d'utiliser :

- Des **blocs horodateurs**, pour la programmation d'actions à des moments prédéfinis ou calculés.
- Une fonctionnalité d'**horodatage**, pour la consignation des durées et des calendriers d'événements et la mesure de la durée de ces derniers.

Pour accéder à l'horloge Date/Heure Twido, sélectionnez **Blocs horodateurs** dans le menu **Logiciel** de TwidoSoft. Notez que cette horloge peut également être réglée à l'aide d'un programme. En cas d'extinction de l'automate, les réglages de l'horloge sont conservés en mémoire pendant un maximum de 30 jours, si la batterie de l'automate était en charge pendant les six heures qui ont précédé l'extinction de l'automate.

L'affichage de l'horloge Date/Heure se fait au format « 24 heures » et tient compte des années bissextiles.

Valeur de correction du RTC

La définition de la valeur de correction du RTC est nécessaire au bon fonctionnement de l'horodateur (ou RTC). Chaque horodateur possède sa propre valeur de correction, figurant au sein même de l'unité. Pour configurer cette valeur dans TwidoSoft, sélectionnez l'option **Configurer RTC** dans la boîte de dialogue **Actions automate**.

Blocs horodateurs

Introduction

Les blocs horodateurs permettent de programmer et de contrôler des actions selon un calendrier précis (mois, jour et heure). Un maximum de 16 blocs horodateurs peuvent être programmés. Ces blocs ne requièrent aucune saisie programme.

Note : Vérifiez le bit système %S51 afin de vous assurer que l'option horodateur (RTC) est installée. Reportez-vous à la rubrique *Bits système (%S), p. 332*. L'option RTC est requise pour l'utilisation de blocs horodateurs.

Paramètres

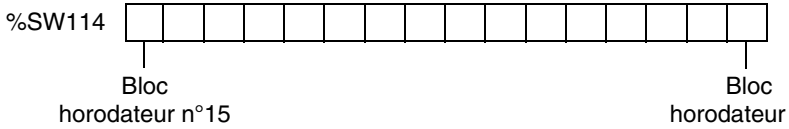
Le tableau suivant répertorie les paramètres d'un bloc horodateur :

Paramètre	Format	Fonction/Plage
Numéro du bloc horodateur	n	n = 0 à 15
Configuré	Case à cocher	Cochez cette case pour configurer le numéro du bloc horodateur sélectionné.
Bit de sortie	%Qx.y.z	L'affectation de la sortie est activée par le bloc horodateur : %Mi ou %Qj.k. Cette sortie est réglée sur 1 lorsque les paramètres de date et d'heure courants sont compris entre les paramètres de début et de fin de la période active.
Mois de début	janvier à décembre	Mois au cours duquel débute le bloc horodateur.
Mois de fin	janvier à décembre	Mois au cours duquel s'achève le bloc horodateur.
Date de début	1 - 31	Jour au cours duquel débute le bloc horodateur.
Date de fin	1 - 31	Jour au cours duquel s'achève le bloc horodateur.
Heure de début	hh:mn	Heure à laquelle débute le bloc horodateur. Définie par l'heure (0 à 23), suivie des minutes (0 à 59).
Heure d'arrêt	hh:mn	Heure à laquelle s'achève le bloc horodateur. Définie par l'heure (0 à 23), suivie des minutes (0 à 59).
Jour de la semaine	lundi à dimanche	Cases à cocher permettant de définir les jours au cours desquels sera activé le bloc horodateur.

Activation de blocs horodateurs

Les bits du mot système %SW114 activent (lorsqu'ils sont réglés sur 1) ou désactivent (lorsqu'ils sont réglés sur 0) le fonctionnement des 16 blocs horodateurs.

Affectation des blocs horodateurs dans %SW114 :



Par défaut (ou après un démarrage à froid), tous les bits de ce mot système sont réglés sur 1. L'utilisation de ces bits par le programme est optionnelle.

Sortie des blocs horodateurs

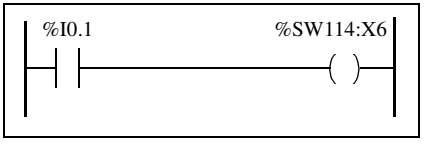
Si la même sortie (%Mi ou %Qj.k) est affectée par plusieurs blocs, l'opérande OR des résultats de chacun des blocs est finalement affecté à cet objet (notez que la même sortie peut disposer de plusieurs « plages de fonctionnement »).

Exemple

Le tableau suivant présente les paramètres d'un programme estival fictif :

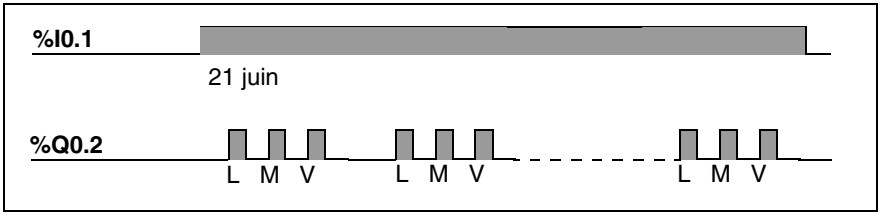
Paramètre	Valeur	Description
Bloc horodateur	6	Bloc horodateur numéro 6
Bit de sortie	%Qx.y.z	Activer la sortie %Qx.y.z
Mois de début	Juin	Débuter l'activité en juin
Mois de fin	Septembre	Arrêter l'activité en septembre
Date de début	21	Débuter l'activité le 21ème jour de juin
Date de fin	21	Arrêter l'activité le 21ème jour de septembre
Jour de la semaine	lundi, mercredi, vendredi	Exécuter l'activité les lundis, mercredis et vendredis
Heure de début	21:00	Débuter l'activité à 21:00
Heure d'arrêt	22:00	Arrêter l'activité à 22:00

Le programme suivant permet de désactiver le bloc horodateur grâce à un commutateur ou un détecteur d'humidité lié à l'entrée %I0.1.




```
LD    %I0.1
ST    %SW114:X6
```

Le chronogramme suivant illustre l'activation de la sortie %Q0.2.



Horodatage par programme

Les paramètres de date et d'heure sont disponibles dans les mots système %SW50 à %SW53 (reportez-vous à la rubrique *Mots système (%SW), p. 341*). Il est ainsi possible d'effectuer un horodatage dans le programme de l'automate en effectuant des comparaisons arithmétiques entre la date et l'heure courantes et les valeurs immédiates ou les mots %MWi (ou %KWi), qui peuvent contenir des consignes.

Horodatage

Introduction

Les mots système %SW50 à %SW53 contiennent les paramètres de date et d'heure au format BCD (reportez-vous à la rubrique *Révision du code BCD, p. 262*), ce qui est utile pour l'affichage sur un périphérique ou la transmission vers ce périphérique. Ces mots système peuvent être utilisés pour stocker les paramètres de date et d'heure d'un événement (reportez-vous à la rubrique *Mots système (%SW), p. 341*.

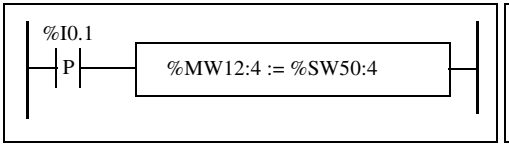
Note : Les paramètres de date et d'heure peuvent également être réglés à l'aide de l'afficheur optionnel (reportez-vous à la rubrique *Horloge Date/Heure, p. 149*).

Datage d'un événement

Pour dater un événement, il suffit d'utiliser des opérations d'affectation, de transférer le contenu de mots système vers des mots internes et de traiter ces mots internes (par exemple, la transmission vers l'afficheur à l'aide de l'instruction EXCH).

Exemple de programmation

L'exemple suivant montre comment dater un front montant sur l'entrée %I0.1.




```
LDR     %I0.0
[%MW12:4 := %SW50:4]
```

Dès qu'un événement est détecté, la table de mots contient :

Codage	Octet le plus significatif	Octet le moins significatif
%MW12	Seconde	Jour de la semaine (1)
%MW13	Heure	Minute
%MW14	Mois	Jour
%MW15	Siècle	Année

Note : (1) 0 = lundi, 1 = mardi, 2 = mercredi, 3 = jeudi, 4 = vendredi, 5 = samedi, 6 = dimanche.

Exemple de table de mots

Exemple de données pour le lundi 19 avril 2002, à 13:40:30 :

Mot	Valeur (hexa.)	Signification
%MW12	3000	30 secondes, 00 = lundi
%MW13	1340	13 heures, 40 minutes
%MW14	0419	04 = avril, le 19
%MW15	2002	2002

Date et heure du dernier arrêt

Les mots système %SW54 à %SW57 contiennent les paramètres de date et d'heure du dernier arrêt et le mot %SW58 contient le code affichant la cause du dernier arrêt, au format BCD (reportez-vous à la rubrique *Mots système (%SW), p. 341*).

Réglage de la date et de l'heure

Introduction

Pour mettre à jour les paramètres de date et d'heure, vous pouvez utiliser l'une des méthodes suivantes :

- TwidoSoft
  Utilisez la boîte de dialogue **Mise à l'heure**. Cette boîte de dialogue est accessible depuis la boîte de dialogue **Actions automate**. Pour afficher cette boîte de dialogue, sélectionnez **Actions automate** dans le menu**Automate** (reportez-vous au guide d'exploitation TwidoSoft).
- Mots système
  Utilisez les mots système %SW50 à %SW53 ou le mot système %SW59.

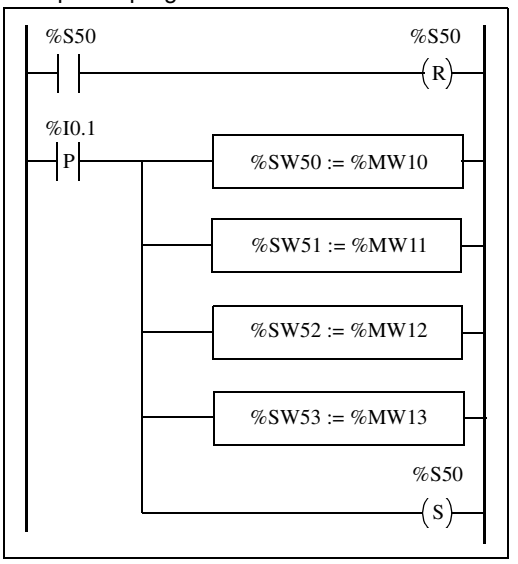
Les paramètres de date et d'heure ne peuvent être mis à jour que si la cartouche optionnelle de l'horodateur (TWDXCPRTC) est installée dans l'automate.

Utilisation des mots %SW 50 à %SW53

Pour utiliser les mots système %SW50 à %SW53 afin de régler la date et l'heure, le bit %S50 doit être réglé sur 1. Ce réglage a les conséquences suivantes :

- La mise à jour des mots %SW50 à %SW53 par l'intermédiaire de l'horloge interne est annulée.
- Les valeurs écrites dans les mots %SW50 à %SW53 sont transférées à l'horloge interne.

Exemple de programmation :




```
LD      %S50
R       %S50

LDR     %I0.1
[%SW50 := %MW10]
[%SW51 := %MW11]
[%SW52 := %MW12]
[%SW53 := %MW13]
S       %S50
```

Les mots %MW10 à %MW13 contiendront les nouveaux paramètres de date et d'heure au format BCD (voir section *Révision du code BCD, p. 262*) et correspondront au codage des mots %SW50 à 53.
La table de mots doit contenir les nouveaux paramètres de date et d'heure.

Codage	Octet le plus significatif	Octet le moins significatif
%MW10	Seconde	Jour de la semaine (1)
%MW11	Heure	Minute
%MW12	Mois	Jour
%MW13	Siècle	Année

Note : (1) 0 = lundi, 1 = mardi, 2 = mercredi, 3 = jeudi, 4 = vendredi, 5 = samedi, 6 = dimanche.

Exemple de données pour le lundi 19 avril 2002 :

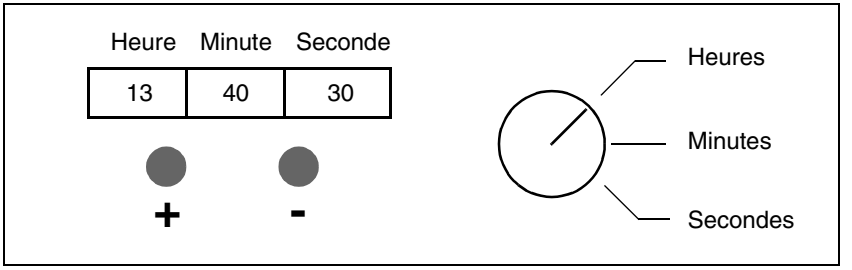
Mot	Valeur (hexa.)	Signification
%MW10	3000	30 secondes, 00 = lundi
%MW11	1340	13 heures, 40 minutes
%MW12	0419	04 = avril, le 19
%MW13	2002	2002

Utilisation du mot %SW59

Pour mettre à jour la date et l'heure, vous pouvez également utiliser le bit système %S59 et le mot système %SW59.
Le réglage du bit %S59 sur 1 permet de régler les paramètres de date et de d'heure courants à l'aide du mot %SW59 (voir section *Mots système (%SW), p. 341*). Le mot système %SW59 permet d'incrémenter ou de décrémenter chacun des composants de date et d'heure sur un front montant.

Exemple de mise en oeuvre

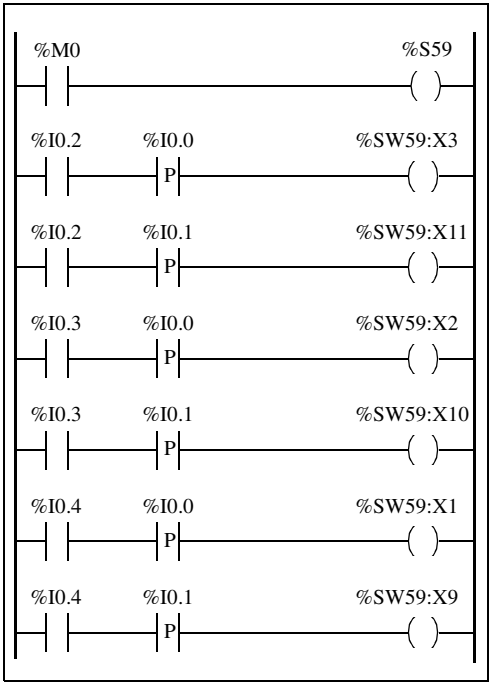
Le panneau avant suivant permet de modifier le réglage de l'horloge interne (heures, minutes et secondes).



Description des commandes :

- Le commutateur Heures/Minutes/Secondes permet de sélectionner l'heure à modifier, respectivement à l'aide des entrées %I0.2, %I0.3 et %I0.4.
- La touche + permet d'incrémenter l'affichage de l'heure sélectionnée, à l'aide de l'entrée %I0.0.
- La touche - permet de décrémenter l'affichage de l'heure sélectionnée, à l'aide de l'entrée %I0.1.

Le programme suivant lit les entrées du panneau et règle l'horloge interne.




```
LD      %M0
ST      %S59
LD      %I0.2          (Heure)
ANDR    %I0.0
ST      %SW59:X3
LD      %I0.2
ANDR    %I0.1
ST      %SW59:X11
LD      %I0.3          (Minute)
ANDR    %I0.0
ST      %SW59:X2
LD      %I0.3
ANDR    %I0.1
ST      %SW59:X10
LD      %I0.4          (Seconde)
ANDR    %I0.0
ST      %SW59:X1
LD      %I0.4
ANDR    %I0.1
ST      %SW59:X9
```

Bits système et mots système

14

En bref...

Présentation

Ce chapitre offre une présentation des bits système et des mots systèmes pouvant être utilisés lors de la création des programmes de régulation d'automates Twido.

Contenu de ce chapitre

Ce chapitre contient les sujets suivants :

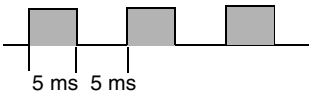
Bits système (%S)

Introduction

La section suivante offre des informations détaillées sur la fonction des bits systèmes, ainsi que sur leur mode de contrôle.

Description détaillée

Le tableau suivant présente une description des bits système, ainsi que leur mode de contrôle.

Bit système	Fonction	Description	Etat initial	Contrôle
%S0	Démarrage à froid	Normalement réglé sur 0, ce bit est réglé sur 1 par : • une reprise de l'alimentation avec perte de données (défaillance d'une batterie) • le programme utilisateur ou l'éditeur de tables d'animation • l'afficheur Ce bit est réglé sur 1 au cours de la première scrutation. Il est ensuite remis à zéro par le système avant la scrutation suivante.	0	S ou U->S
%S1	Démarrage à chaud	Normalement réglé sur 0, ce bit est réglé sur 1 par : • une reprise de l'alimentation avec enregistrement des données • le programme utilisateur ou l'éditeur de tables d'animation • l'afficheur Il est ensuite remis à zéro par le système une fois la scrutation terminée.	0	S ou U->S
%S4 %S5 %S6 %S7	Base temps : 10 ms Base temps : 100 ms Base temps : 1 s Base temps : 1 min	Les modifications d'état de ces bits sont régulées par une horloge interne. Ces modifications ne sont pas synchronisées avec la scrutation de l'automate. Exemple : %S4  	-	S

Bit système	Fonction	Description	Etat initial	Contrôle
%S8	Gel de la sortie	Initialement réglé sur 1, ce bit peut être réglé sur 0 par le programme ou par le terminal (dans l'éditeur de tables d'animations) : • A l'état 1, efface les sorties pendant l'état NO CONFIG. • A l'état 0, autorise les tests de liaison pendant l'état NO CONFIG.	1	U
%S9	Remise à zéro des sorties	Normalement non réglé. Ce bit peut être réglé sur 1 par le programme ou par le terminal (dans l'éditeur de tables d'animations) : • A l'état 1, force la valeur des sorties sur 0 lorsque l'automate est en mode d'exécution (RUN). • A l'état 0, les sorties sont mises à jour normalement.	0	U
%S10	Défaillance d'E/S	Normalement réglé sur 1. Réglé sur 0 par le système lorsqu'une défaillance d'E/S est détectée.	1	S
%S11	Débordement du chien de garde	Normalement réglé sur 0. Réglé sur 1 par le système lorsque la durée d'exécution du programme (durée de scrutation) dépasse la durée de scrutation maximale (chien de garde logiciel). Le débordement du chien de garde fait passer l'automate en mode HALT.	0	S
%S12	Automate en cours d'exécution	Ce bit reflète l'état d'exécution de l'automate. Le système règle le bit sur 1 lorsque l'automate est en cours d'exécution. A l'arrêt, lors de l'initialisation du système et en tout autre état, ce bit est réglé sur 0.	0	S
%S13	Première scrutation	Normalement réglé sur 0, ce bit est réglé sur 1 par le système au cours de la première scrutation après que l'automate est passé en mode exécution (RUN).	1	S

Bit système	Fonction	Description	Etat initial	Contrôle
%S17	Dépassement	Normalement réglé sur 0, ce bit est réglé sur 1 par le système : • en cas de dépassement au cours d'une opération arithmétique sans signe (reste). • au cours d'une opération de rotation ou de décalage. Le système règle la sortie d'un bit sur 1. Doit être testé par le programme utilisateur après chaque opération pouvant provoquer un débordement, puis remis à zéro par l'utilisateur en cas de débordement.	0	S->U
%S18	Débordement ou erreur arithmétique	Normalement réglé sur 0. Réglé sur 1 en cas de débordement découlant de l'exécution d'une opération de 16 bits générant : • un résultat supérieur à + 32 767 ou inférieur à - 32 768 • une division par 0 • la racine carrée d'un nombre négatif • une conversion BTI ou ITB non significative : valeur BCD hors plage Doit être testé par le programme utilisateur après chaque opération pouvant provoquer un débordement, puis remis à zéro par l'utilisateur en cas de débordement.	0	S->U
%S19	Débordement de la période de scrutation (scrutation périodique)	Normalement réglé sur 0, ce bit est réglé sur 1 par le système en cas de débordement d'une période de scrutation (durée de scrutation supérieure à la durée définie par l'utilisateur au moment de la configuration, ou programmée dans %SW0). Ce bit est remis à zéro par l'utilisateur.	0	S->U

Bit système	Fonction	Description	Etat initial	Contrôle
%S20	Dépassement d'index	Normalement réglé sur 0, ce bit est réglé sur 1 lorsque le repère de l'objet indexé devient inférieur à 0 ou supérieur à la taille maximale d'un objet. Doit être testé par le programme utilisateur après chaque opération pouvant provoquer un débordement, puis remis à zéro en cas de débordement.	0	S->U
%S21	Initialisation de GRAFCET	Normalement réglé sur 0, ce bit est réglé sur 1 par : ● un redémarrage à froid, %S0=1 ● le programme utilisateur, uniquement dans la section du programme de prétraitement, à l'aide de l'instruction Set (S %S21) ou d'une bobine Set -(S)- %S21 ● le terminal A l'état 1, il provoque l'initialisation de GRAFCET. Tous les pas actifs sont désactivés et les pas initiaux sont activés. Il est ensuite remis à zéro par le système après l'initialisation de GRAFCET.	0	U->S
%S22	GRAFCET RESET	Normalement réglé sur 0, ce bit ne peut être réglé sur 1 par le programme qu'au cours du prétraitement. A l'état 1, il provoque la désactivation des pas de l'ensemble du GRAFCET. Il est remis à zéro par le système au début de l'exécution du traitement séquentiel.	0	U->S

Bit système	Fonction	Description	Etat initial	Contrôle
%S23	Préréglage et gel de GRAFCET	Normalement réglé sur 0, ce bit ne peut être réglé sur 1 par le programme que dans le module du programme de prétraitement. A l'état 1, il valide le préréglage du graphique GRAFCET. Le maintien de ce bit sur la valeur 1 a pour effet de geler le GRAFCET (gel du graphique). Il est remis à zéro par le système au début de l'exécution du traitement séquentiel pour garantir que le graphique GRAFCET pourra sortir de la situation de gel.	0	U->S
%S24	Afficheur	Normalement réglé sur 0, ce bit peut être réglé sur 1 par l'utilisateur. ● A l'état 0, l'afficheur fonctionne normalement. ● A l'état 1, l'afficheur est gelé, conserve l'affichage courant, le clignotement est désactivé et le traitement des saisies s'interrompt.	0	U->S
%S50	Mise à jour de la date et de l'heure à l'aide des mots %SW50 à %SW53	Normalement réglé sur 0, ce bit peut être réglé sur 1 ou sur 0 par le programme ou l'afficheur. ● A l'état 0, la date et l'heure peuvent être lues. ● A l'état 1, la date et l'heure peuvent être mises à jour.	0	U->S

Bit système	Fonction	Description	Etat initial	Contrôle
%S51	Etat de l'horloge Date/Heure	Normalement réglé sur 0, ce bit peut être réglé sur 1 ou sur 0 par le programme ou l'afficheur. • A l'état 0, la date et l'heure sont réglées. • A l'état 1, la date et l'heure doivent être réglées par l'utilisateur. Lorsque ce bit est réglé sur 1, les données de l'horloge Date/Heure ne sont pas valides. Il est possible que la date et l'heure n'aient jamais été configurées, que le niveau de la batterie soit faible ou que la constante de correction de l'automate ne soit pas valide. Le passage de l'état 1 à 0 force l'écriture de la constante de correction sur l'horodateur.	0	U->S
%S59	Mise à jour de la date et de l'heure à l'aide du mot %SW59	Normalement réglé sur 0, ce bit peut être réglé sur 1 ou sur 0 par le programme ou l'afficheur. • A l'état 0, la date et l'heure restent inchangées. • A l'état 1, la date et l'heure sont incrémentées ou décrémentées en fonction des bits de contrôle réglés dans %SW59.	0	U
%S69	Affichage STAT LED utilisateur	A l'état 0, STAT LED est désactivé. A l'état 1, STAT LED est activé.	0	U
%S70	Rafraîchissement des données sur le bus AS-i	Ce bit est réglé sur 1 par le système à la fin de chaque cycle de l'automate ou à la fin du cycle de scrutation du bus AS-i. A la mise sous tension, ce bit indique que toutes les données ont été rafraîchies au moins une fois et qu'elles sont par conséquent significatives. Ce bit doit être remis à zéro par l'utilisateur.	0	S->U

Bit système	Fonction	Description	Etat initial	Contrôle
%S73	Basculement en mode protégé sur le bus AS-i	Normalement réglé sur 0, ce bit est réglé sur 1 par l'utilisateur afin de basculer en mode protégé sur le bus AS-i. Avant cela, le bit doit avoir été préalablement réglé sur 1. Ce bit ne sera utilisé que lors du test d'un système de liaison. Il n'a aucune utilité au sein de l'automate.	0	S
%S74	Enregistrement de la configuration du bus AS-i	Normalement réglé sur 0, ce bit est réglé sur 1 par l'utilisateur afin de permettre l'enregistrement de la configuration courante dans le bus AS-i. Ce bit ne sera utilisé que lors du test d'un système de liaison. Il n'a aucune utilité au sein de l'automate.	0	S
%S96	Programme de sauvegarde OK	Ce bit peut être lu à n'importe quel moment (soit par le programme ou lors d'un réglage), en particulier après un démarrage à froid ou un redémarrage à chaud. ● A l'état 0, le programme de sauvegarde n'est pas valide. ● A l'état 1, le programme de sauvegarde est valide.	0	S
%S97	Enregistrer %MW OK	Ce bit peut être lu à n'importe quel moment (soit par le programme ou lors d'un réglage), en particulier après un démarrage à froid ou un redémarrage à chaud. ● A l'état 0, l'enregistrement %MW n'est pas OK. ● A l'état 1, l'enregistrement %MW est OK.	0	S
%S100	Raccordement du câble de communications TwidoSoft	Indique si le câble de communications TwidoSoft est raccordé. ● A l'état 1, soit le câble de communications TwidoSoft n'est pas raccordé, soit TwidoSoft est connecté. ● A l'état 0, le câble de liaison distante TwidoSoft est raccordé.	-	S

Bit système	Fonction	Description	Etat initial	Contrôle
%S110	Echanges de liaison distante	Ce bit est remis à zéro par le programme ou par le terminal. • A l'état 1 pour un maître, tous les échanges de liaison distante (E/S distante uniquement) sont terminés. • A l'état 1 pour un esclave, l'échange avec maître est terminé.	0	S->U
%S111	Echange de liaison distante unique	• A l'état 0 pour un maître, un échange de liaison distante unique est terminé. • A l'état 0 pour un esclave, un échange de liaison distante unique est détecté. • A l'état 1 pour un maître, un échange de liaison distante unique est actif. • A l'état 1 pour un esclave, un échange de liaison distante unique est détecté.	0	S
%S112	Connexion de liaison distante	• A l'état 0 pour un maître, la liaison distante est désactivée. • A l'état 1 pour un maître, la liaison distante est activée.	0	U
%S113	Configuration/ fonctionnement de la liaison distante	• A l'état 0 pour un maître ou un esclave, la configuration/le fonctionnement de la liaison distante est OK. • A l'état 1 pour un maître, la configuration ou le fonctionnement de la liaison distante comporte une erreur. • A l'état 1 pour un esclave, la configuration ou le fonctionnement de la liaison distante comporte une erreur.	0	S->U
%S118	Erreur d'E/S distante	Normalement réglé sur 1. Réglé sur 0 lorsqu'une défaillance d'E/S est détectée sur le lien distant.	1	S

Bit système	Fonction	Description	Etat initial	Contrôle
%S119	Erreur d'E/S locale	Normalement réglé sur 1. Réglé sur 0 lorsqu'une défaillance d'E/S est détectée sur l'E/S locale (base ou expansion). %SW118 détermine la nature de la défaillance. Remis à 1 lorsque la défaillance est résolue.	1	S

Description des abréviations utilisées dans le tableau précédent

Abréviation	Description
S	Contrôlé par le système
U	Contrôlé par l'utilisateur
U->S	Réglé sur 1 par l'utilisateur, remis à zéro par le système
S->U	Réglé sur 1 par le système, remis à zéro par l'utilisateur

Mots système (%SW)

Introduction

La section suivante offre des informations détaillées sur la fonction des mots système, ainsi que sur leur mode de contrôle.

Description détaillée

Le tableau suivant offre des informations détaillées sur la fonction des mots système, ainsi que sur leur mode de contrôle.

Mots système	Fonction	Description	Contrôle
%SW0	Période de scrutation de l'automate (tâche périodique)	Modifie la période de scrutation de l'automate, définie au moment de la configuration à l'aide du programme utilisateur dans l'éditeur de tables d'animation.	U
%SW6	Etat de l'automate	Etat de l'automate : 0 = NO CONFIG 2 = STOPPED 3 = RUN 4 = HALT	S

Mots système	Fonction	Description	Contrôle
%SW7	Etat de l'automate	Bit [0] Sauvegarde/restauration en cours Bit [1] Configuration de l'automate OK Bit [3..2] Bits d'état EEPROM : • 00 = Pas de cartouche • 01 = Cartouche EEPROM 32 Ko • 10 = Cartouche EEPROM 64 Ko • 11 = Réservé à une utilisation ultérieure Bit [4] Application RAM différente d'EEPROM (1 = oui) Bit [5] Application RAM différente de cartouche (1 = oui) Bit [6] Certaines tâches de périphériques sont en mode STOP Bit [7] Automate réservé Bit [8] Application en mode Ecriture protégée Bit [9] Non utilisé Bit [10] Second port série installé Bit [11] Type du second port série (0 = EIA RS-232, 1 = EIA RS-485) Bit [12] Application valide en mémoire interne (1 = oui) Bit [13] Application valide en cartouche (1 = oui) Bit [14] Application valide en RAM (1 = oui) Bit [15] Prêt pour exécution	S
%SW11	Heure du chien de garde logiciel	Initialise l'heure maximale du chien de garde. La valeur (10 à 500 ms) est définie par la configuration.	U
%SW18-%SW19	Compteur de temporisateur absolu 100 ms	Compteur de temporisateur absolu 100 ms. **%SW18** correspond aux octets les moins significatifs et **%SW19**, aux octets les plus significatifs du mot double.	S et U
%SW30	Durée de la dernière scrutation	Affiche la durée d'exécution du dernier cycle de scrutation de l'automate (en ms). **Note :** Cette durée correspond au temps qui s'écoule entre le début (acquisition des entrées) et la fin (mise à jour des sorties) d'un cycle de scrutation.	S

Mots système	Fonction	Description	Contrôle
%SW31	Durée de scrutation max.	Affiche la durée d'exécution du plus long cycle de scrutation de l'automate (en minutes), depuis le dernier démarrage à froid. **Note :** Cette durée correspond au temps qui s'écoule entre le début (acquisition des entrées) et la fin (mise à jour des sorties) d'un cycle de scrutation.	S
%SW32	Durée de scrutation min.	Affiche la durée d'exécution du cycle de scrutation de l'automate le plus court (en minutes), depuis le dernier démarrage à froid. **Note :** Cette durée correspond au temps qui s'écoule entre le début (acquisition des entrées) et la fin (mise à jour des sorties) d'un cycle de scrutation.	S

Mots système	Fonction	Description		Contrôle
%SW49 %SW50 %SW51 %SW52 %SW53	Fonction de bloc horodateur	Fonction de bloc horodateur (RTC) mots contenant les valeurs de date et d'heure courantes (en BCD) :		S et U
		%SW49	xN jour de la semaine (N=0 pour lundi)	
		%SW50	00SS Secondes	
		%SW51	HHMM Heure et minutes	
		%SW52	MMDD Mois et jour	
		%SW53	CCYY Siècle et année	
		Ces mots sont contrôlés par le système lorsque le bit **%S50** est réglé sur 0. Ces mots peuvent être écrits par le programme utilisateur ou par le terminal, lorsque le bit **%S50** est réglé sur 1.		
%SW54 %SW55 %SW56 %SW57	Fonction de bloc horodateur	Fonction de bloc horodateur (RTC). Mots système contenant la date et l'heure de la dernière coupure secteur ou du dernier arrêt de l'automate (en BCD) :		S
		%SW54	SS Secondes	
		%SW55	HHMM Heure et minutes	
		%SW56	MMDD Mois et jour	
		%SW57	CCYY Siècle et année	

Mots système	Fonction	Description		Contrôle
%SW58	Code du dernier arrêt	Affiche le code indiquant la cause du dernier arrêt :		S
		1 =	Front de l'entrée Run/ Stop	
		2 =	Arrêt en cas de défaillance logicielle (débordement de la scrutation de l'automate)	
		3 =	Commande d'arrêt	
		4 =	Coupure secteur	
		5 =	Arrêt en cas de défaillance matérielle	

Mots système	Fonction	Description			Contrôle
%SW59	Régler la date courante	Règle la date courante. Contient deux jeux de 8 bits permettant de régler la date courante. L'opération est toujours effectuée sur le front montant du bit. Ce mot est activé par le bit **%S59**.			U
		Incrément	**Décrément**	**Paramètre**	
		bit 0	bit 8	Jour de la semaine	
		bit 1	bit 9	Secondes	
		bit 2	bit 10	Minutes	
		bit 3	bit 11	Heures	
		bit 4	bit 12	Jours	
		bit 5	bit 13	Mois	
		bit 6	bit 14	Années	
		bit 7	bit 15	Siècles	
%SW60	Valeur de correction du RTC	Valeur de correction du RTC			U

Mots système	Fonction	Description	Contrôle
%SW63	Code d'erreur du bloc EXCH1	Si une erreur survient lors de l'utilisation du bloc EXCH, les bits de sortie %MSG.D et %MSG.E passent sur 1. Ce mot système contient le code de l'erreur. Les valeurs possibles sont les suivantes : ● 0 : Pas d'erreur, échange correct ● 1 : Tampon de transmission trop grand ● 2 : Tampon de transmission trop petit ● 3 : Tableau trop petit ● 4: Réception débordement table ● 5 : Délai écoulé ● 6 : Erreur de transmission ● 7 : Mauvaise commande ASCII (mode ASCII uniquement) ● 8 : Port sélectionné non configurable/disponible ● 9 : Erreur de réception (mode ASCII uniquement) ● 10 : Tableau %KWi interdit ● 11: Offset transmission plus large que la table de transmission ● 12: Offset réception plus large que la table de réeception ● 13: Traitement EXCH arrête par l'automate Ce mot est réglé sur 0 chaque fois que le bloc EXCH est utilisé.	S
%SW64	Code d'erreur du bloc EXCH2	Identique à %SW63	S
%SW67	Fonction et type d'automate	Contient les informations suivantes : ● Bits de type d'automate [0 -11] ● 8B0 = TWDLCAA10DRF ● 8B1 = TWDLCAA16DRF ● 8B2 = TWDLMDA20DUK/DTK ● 8B3 = TWDLCAA24DRF ● 8B4 = TWDLMDA40DUK/DTK ● 8B6 = TWDLMDA20DRT ● Bit 12 non utilisé = 0 ● Bits de repère de liaison distante [13-15] ● 000 = automate maître ● 001 - 111 = automate distant 1-7 ● 001 = repère 1 ● 111 = repère 7	S

Mots système	Fonction	Description	Contrôle
%SW76 à %SW79	Décompteurs 1-4	Ces quatre mots sont utilisés comme des temporisateurs à 1 ms. Ces mots sont décrémentés de manière individuelle par le système, toutes les millisecondes, s'ils possèdent une valeur positive. Ceci donne quatre décompteurs comptant en ms, égaux à une plage de fonctionnement de 1 à 32 767 ms. Le réglage du bit 15 sur 1 permet d'interrompre la décrémentation.	S and U
%SW96	Command et/ ou diagnostics de la fonction de sauvegarde/ restauration de la'application et %MW.	Bit [0] Ce bit est défini par la logigue utilisateur afin de spécifier l'enregistrement des mots mémoire %MWi dans l'EEPROM. Le programme réinitialise ce bit lorsque l'enregistrement de %MW a démarré, et non pasà la fin du processus. Bit [1] Ce bit est défini par le microprogramme et indique la fin de l'enregistrement. Ce qui signifie que lorsque le bit est sur 1, toute demande d'enregistrement dans l'EEPROM a été effectuée. Il est redéfini sur zéro àla prochaine demande de sauvegarde dans l'EEPROM. Bit [2] Lorsqu'il est défini sur 1, il indique qu'une erreur s'est produite lors de la derniére demande de sauvegarde ou de restauration des paramètres. Pour plus d'informations, voir les bits 8, 9, 10 et 14. Bit [6] L'automate contient une application valide (1 = oui). Bit [8] Le nombre de %MW spécifié dans %SW97 est supérieur au nombre maximal configurédans l'application utilisateur (1 = oui). Bit [9] Le nomber de %MW spécifiédans %SW97 est supérieur au nombre maximal Bit [10] Différence entre la RAM et l'EEPROM interne (1 = oui). Bit [14] Erreur d'écriture EEPROM (1 = oui)	

Mots système	Fonction	Description	Contrôle
%SW97	Commande ou diagnostics de fonction d'enregistrement et de restauration	Cette valeur correspond au nombre physique de mots système **%MW**, devant être uniquement enregistrés en EEPROM interne. Elle n'est pas utilisée pour restaurer des mots système. Lorsque cette valeur est réglée sur 0, les mots mémoire ne sont pas stockés. L'utilisateur doit définir le programme de logique utilisateur. Dans le cas contraire, ce programme sera réglé sur 0 dans l'application de l'automate, sauf dans le cas suivant : Lors d'un démarrage à froid, ce mot est réglé sur -1 si l'EEPROM Flash interne ne possède pas de fichier de mot mémoire **%MW** enregistré. Lors d'un démarrage à froid au cours duquel l'EEPROM Flash interne contient un fichier de mot mémoire **%MW**, la valeur du nombre de mots mémoire enregistrés dans le fichier doit être réglé dans le mot système **%SW97**.	U

Mots système	Fonction	Description	Contrôle
%SW111	Etat de la liaison distante	Deux bits pour chaque automate distant (uniquement maître) : x0-5:0 - automate distant 1-6 non présent 1- automate distant 1-6 présent x6:0 - automate distant 7 non présent 1- automate distant 7 présent x8-13:0 - E/S distante détectée sur l'automate distant 1-6 1 - automate d'extension détecté sur l'automate distant 1-6 x14:0 - E/S distante détectée sur l'automate distant 7 1 - automate d'extension détecté sur l'automate distant 7	S

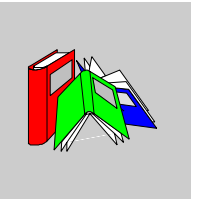
Mots système	Fonction	Description	Contrôle
%SW112	Code d'erreur de configuration ou de fonctionnement de la liaison distante	0 - opérations réussies 1 - expiration du délai (esclave) 2 - erreur de checksum détectée (esclave) 3 - incohérence de configuration (esclave) Ceci est défini par le système et doit être redéfini par l'utilisateur.	S
%SW113	Configuration de la liaison distante	Deux bits pour chaque automate distant (uniquement maître) : x0-5:0 - automate distant 1-6 non configuré 1- automate distant 1-6 configuré x6:0 - automate distant 7 non configuré 1- automate distant 7 configuré x8-13:0 - E/S distante configurée en tant qu'automate distant 1-6 1 - automate d'extension configuré en tant qu'automate distant 1-6 x14:0 - E/S distante détectée en tant qu'automate distant 7 1 - automate d'extension configuré en tant qu'automate distant7	S
%SW114	Activer les blocs horodateurs (RTC)	Active ou désactive le fonctionnement des blocs horodateurs (RTC), par l'intermédiaire du programme utilisateur ou de l'afficheur. Bit 0 : 1 = active le bloc horodateur n°0 Bit 15 : 1 = active le bloc horodateur n°15 Tous les blocs horodateurs sont initialement activés, l'état initial est 0. Si aucun bloc horodateur n'est configuré, la valeur par défaut est FFFF.	S et U
%SW118	Mot d'état de la base automate	Affiche les défaillances détectées sur l'automate maître. Bit 9 : 0 = défaillance ou comm. externe Défaillance Bit 12 : 0 = horodateur (RTC) non installé Bit 13 : 0 = défaillance de configuration (extension d'E/S configurée, mais absente ou défaillante). Tous les autres bits de ce mot sont réglés sur 1 et sont réservés. Pour un automate ne présentant aucune défaillance, la valeur de ce mot est FFFFh.	S

Mots système	Fonction	Description	Contrôle
%SW120	Fonctionnement des modules d'E/S d'expansion	Un bit par module. Repère 0 = Bit 0 1 = Mauvaise condition 0 = OK	S

Description des abréviations utilisées dans le tableau précédent

Abréviation	Description
S	Contrôlé par le système
U	Contrôlé par l'utilisateur

Glossaire



!

% Préfixe qui identifie les repères de mémoire interne utilisés dans l'automate pour stocker les valeurs des variables, les constantes, les E/S, etc., du programme.

A

Afficheur de références croisées Fenêtre spécialisée de l'application TwidoSoft permettant de visualiser les références croisées.

Afficheur des erreurs du programme Fenêtre TwidoSoft spécialisée permettant d'afficher les avertissements et erreurs du programme.

Analyser le programme Commande permettant de compiler un programme et de rechercher les erreurs qu'il pourrait contenir : erreurs de syntaxe et de structure, symboles sans repère correspondant, ressources non disponibles que le programme tente d'utiliser, et taille de programme trop importante pour la capacité de mémoire de l'automate. Les erreurs sont répertoriées dans l'afficheur des erreurs du programme.

Application Une application TwidoSoft est composée d'un programme, de données de configuration, de symboles et d'une documentation.

Arrêter Commande permettant d'arrêter un programme d'application exécuté par l'automate.

ASCII De l'anglais American Standard Code for Information Interchange. Protocole de communication utilisant sept bits pour représenter les caractères alphanumériques, notamment les lettres, les nombres et certains caractères graphiques et de contrôle.

Automate Automate programmable Twido. Il existe deux types d'automates : les automates compacts et les automates modulaires.

Automate compact Type d'automate Twido fournissant une configuration simple monobloc avec une expansion limitée. Les automates modulaires constituent l'un des deux types d'automates Twido.

Automate d'extension Automate Twido configuré en tant qu'esclave sur un réseau de liaison distante. Une application peut être exécutée dans la mémoire de l'automate d'extension et le programme peut accéder aux données d'E/S locales et d'expansion, mais les données d'E/S ne peuvent pas être transmises à l'automate maître. Le programme exécuté dans l'automate d'extension transmet ses informations à l'automate maître à l'aide de mots réseau (%INW et QNW).

Automate distant Automate Twido configuré pour communiquer avec un automate maître sur un réseau de liaison distante.

Automate maître Automate Twido configuré en tant que maître sur un réseau de liaison distante.

Automate modulaire Type d'automate Twido offrant une configuration flexible avec des possibilités d'extension. Les automates compacts constituent l'un des deux types d'automates Twido.

Automate programmable Automate Twido. Il existe deux types d'automates : les automates compacts et les automates modulaires.

B

Bloc fonction Unité de programme comportant des entrées et des variables organisées pour calculer des valeurs de sorties à l'aide d'une fonction définie, telle qu'un temporisateur ou un compteur.

Blocs horodateurs Bloc fonction utilisé pour programmer les fonctions de réglage de la date et de l'heure afin de contrôler les événements. Nécessite l'option Horodateur (RTC).

Bobine Elément du schéma à contacts représentant une sortie de l'automate.

Bus d'expansion Permet de connecter les modules d'E/S d'expansion à la base automate.

C

Cartouche de mémoire Les cartouches de sauvegarde de mémoire fournies en option permettent de sauvegarder et de restaurer une application (données de programme et de configuration). Deux tailles sont disponibles : 32 et 64 Ko.

Chargement automatique Fonction constamment activée permettant de transférer automatiquement une application d'une cartouche de sauvegarde vers la RAM de l'automate en cas de perte ou d'altération de l'application. A la mise sous tension, l'automate compare l'application se trouvant dans la RAM de l'automate avec celle de la cartouche de sauvegarde de mémoire en option (si elle est installée). En cas de différence, la copie de la cartouche de sauvegarde est copiée dans l'automate et dans la mémoire EEPROM interne. Si aucune cartouche de sauvegarde n'est installée, l'application de la mémoire EEPROM interne est copiée dans l'automate.

Commentaires Les commentaires sont des messages textuels que vous entrez afin de donner des informations sur la finalité d'un programme. Pour les programmes par schémas à contacts, vous pouvez entrer jusqu'à trois lignes de texte dans l'en-tête réseau pour décrire la finalité du réseau. Chaque ligne peut contenir un maximum de 64 caractères. Pour les programmes par listes, vous pouvez entrer le texte sur une ligne de programme non numérotée. Les commentaires doivent être insérés entre parenthèses et astérisques. Exemple : (*COMMENTAIRES*).

Compteur Bloc fonction utilisé pour compter les événements (comptage ou décomptage).

Compteurs rapides (FC) Bloc fonction proposant une fonction de comptage/décomptage plus rapide que celle du bloc fonction Compteurs. Un compteur rapide (FC) peut compter à une fréquence maximale de 5 kHz.

Compteurs rapides (VFC) Bloc fonction proposant une fonction de comptage plus rapide que celle des blocs fonctions Compteurs et Compteurs rapides (FC). Un compteur rapide (VFC) peut compter à une fréquence maximale de 20 kHz.

Constante Unité de mémoire, telle qu'un bit ou un mot, dont le contenu ne peut pas être modifié par le programme en cours d'exécution.

Contact Elément du schéma à contacts représentant une entrée de l'automate.

D

Démarrage ou redémarrage à froid

Démarrage de l'automate avec toutes les données initialisées sur les valeurs par défaut, le programme démarrant avec toutes les variables effacées. Tous les paramètres logiciels et matériels sont initialisés. Une coupure secteur (automates compacts uniquement) ou le chargement d'une nouvelle application dans la mémoire RAM de l'automate provoquent automatiquement un redémarrage à froid. Le démarrage des automates compacts et des automates sans batterie de sauvegarde est toujours un démarrage à froid.

E

Editeur de configuration

Fenêtre spécialisée de TwidoSoft permettant de gérer les configurations logicielles et matérielles.

Editeur de schémas à contacts

Fenêtre TwidoSoft spécialisée permettant d'éditer un programme par schémas à contacts.

Editeur de tables d'animation

Fenêtre spécialisée de l'application TwidoSoft permettant de visualiser et de créer des tables d'animation.

Editeur List

Simple éditeur de programmes permettant de créer et d'éditer un programme par listes.

EEPROM

Mémoire morte effaçable et programmable électriquement (de l'anglais Electrically Erasable Programmable Read-Only Memory). Twido est doté d'une mémoire EEPROM interne et d'une cartouche de mémoire EEPROM externe en option.

Effacer

Cette commande permet de supprimer le contenu enregistré dans l'application. Elle comporte deux options : supprimer le contenu de la mémoire RAM de l'automate, de la mémoire EEPROM interne de l'automate et de la cartouche de sauvegarde installée en option, ou ne supprimer que le contenu de la cartouche de sauvegarde installée en option.

En-tête réseau

Panneau apparaissant directement sur un réseau de schéma à contacts et pouvant être utilisé pour donner des informations sur la finalité de celui-ci.

Entrée à mémorisation d'état	Les impulsions entrantes sont capturées et enregistrées afin d'être analysées ultérieurement par l'application.
Etape	Une étape Grafcet désigne un état du fonctionnement séquentiel de l'automate.
Etat en ligne	Etat de fonctionnement de TwidoSoft qui est affiché sur la barre d'état lorsqu'un PC est connecté à un automate.
Etat hors ligne	Etat de fonctionnement de TwidoSoft qui est affiché sur la barre d'état lorsque aucun PC n'est connecté à un automate.
Etat initial	Etat de fonctionnement de TwidoSoft affiché sur la barre d'état lorsque TwidoSoft démarre ou qu'aucune application n'est ouverte.
Etats de fonctionnement	Indique l'état de TwidoSoft. Affiché sur la barre d'état. Il existe quatre états de fonctionnement : initial, hors ligne, en ligne et surveillance.
Exécuter	Commande permettant d'exécuter un programme d'application sur l'automate.
Executive Loader	Application Windows 32 bits permettant de télécharger un nouveau microprogramme de l'automate vers un automate Twido.

F

Fichier d'application	Les applications Twido sont enregistrées dans des fichiers avec l'extension .twd.
FIFO	Premier entré, Premier sorti (de l'anglais First In, First Out). Bloc fonction permettant de mettre les opérations en file d'attente.
Fonctionnement en ligne	Mode de fonctionnement de TwidoSoft dans lequel un PC est connecté à l'automate et dans lequel l'application contenue dans la mémoire du PC est identique à celle contenue dans la mémoire de l'automate. Le fonctionnement en ligne permet de déboguer et de régler une application.
Fonctionnement hors ligne	Mode de fonctionnement de TwidoSoft dans lequel aucun PC n'est connecté à l'automate et dans lequel l'application contenue dans la mémoire du PC est différente de celle contenue dans la mémoire de l'automate. Le fonctionnement hors ligne permet de créer et de développer une application.

Fonctions Date/ Heure Permettent de contrôler les événements par mois, jour et heure. Voir "Blocs horodateurs".

Forçage Attribution volontaire des valeurs 0 et 1 aux entrées et sorties de l'automate, même si les valeurs réelles sont différentes. Permet de déboguer un programme pendant son animation.

G

Gestionnaire de ressources Composant de TwidoSoft qui contrôle l'occupation mémoire d'une application lors de la programmation et de la configuration, en suivant les références aux objets logiciels faites par une application. Un objet est considéré comme étant référencé par l'application lorsqu'il est utilisé comme opérande dans une instruction de liste ou dans un réseau de schéma à contacts. Affiche les informations d'état relatives au pourcentage de mémoire totale utilisée et émet un avertissement si l'espace mémoire disponible est insuffisant. Voir "Indicateur d'utilisation de la mémoire".

Grafcet Un programme écrit en langage Grafcet se compose d'étapes contenant une description graphique et structurée du fonctionnement séquentiel d'un automate. Des symboles graphiques simples sont utilisés pour décrire la séquence des étapes.

H

Horodateur Option permettant de maintenir une horloge à l'heure pendant une durée déterminée lorsque l'automate n'est pas sous tension.

I

Indicateur d'utilisation de la mémoire Section de la barre d'état de la fenêtre principale de TwidoSoft qui affiche le pourcentage d'utilisation de la mémoire totale de l'automate par l'application. Emet un avertissement lorsque l'espace mémoire disponible est insuffisant.

Initialiser Commande qui rétablit les états initiaux de toutes les valeurs des données. L'automate doit être en mode Stop ou Error.

Instance Dans un programme, objet unique qui appartient à un type précis de bloc fonction. Par exemple, dans le format de temporisateur %TMi, i est un nombre qui représente l'instance.

Instructions réversibles Méthode de programmation permettant de visualiser les instructions alternativement comme des instructions de liste ou des réseaux de schémas à contacts.

L

Langage à contact Un programme écrit en langage à contact consiste en la représentation graphique d'instructions d'un programme de l'automate, avec des symboles pour les contacts, bobines et blocs, sous la forme d'une série de réseaux exécutés de manière séquentielle par un automate.

Langage liste d'instructions (List) Un programme écrit en langage liste d'instructions (IL) consiste en une série d'instructions exécutées de manière séquentielle par l'automate. Chaque instruction comprend un numéro de ligne, un code d'instruction et un opérande.

Liaison distante Bus maître/esclave à haut débit conçu pour assurer l'échange d'une petite quantité de données entre un automate maître et un maximum de sept automates (esclaves) distants. Deux types d'automates distants peuvent être configurés pour transférer des données vers un automate maître : un automate d'extension pour transférer les données d'application et un automate d'E/S distantes pour transférer les données d'E/S. Un réseau de liaison distante peut comprendre des automates des deux types.

LIFO Dernier entré, Premier sorti (de l'anglais Last In, First Out). Bloc fonction permettant d'effectuer des opérations de pile.

Lignes de commentaire Dans les programmes par listes, les commentaires peuvent être entrés sur des lignes distinctes des instructions. Les lignes de commentaires ne sont pas numérotées. Elles doivent être insérées entre parenthèses et astérisques. Exemple : (*COMMENTAIRES*).

M

Microprogramme de l'automate
Le microprogramme de l'automate est un système d'exploitation qui exécute les applications et gère les opérations de l'automate.

Modbus
Protocole de communication maître-esclave permettant à un maître unique d'obtenir des réponses des esclaves.

Mode de scrutation
Indique la façon dont l'automate scrute un programme. Il existe deux types de modes de scrutation : le mode normal (cyclique), dans lequel la scrutation s'effectue en permanence, ou le mode périodique, dans lequel la scrutation ne s'effectue que pendant une durée limitée (dans une plage de 2 à 150 ms) avant de lancer la scrutation suivante.

Mode Surveillance
Etat de fonctionnement de TwidoSoft qui est affiché sur la barre d'état lorsqu'un PC est connecté à un automate dans un mode sans écriture.

Modules d'E/S d'expansion
Les modules d'E/S d'expansion en option sont disponibles pour ajouter des points d'E/S à un automate Twido. (Certains modèles d'automate ne prennent pas en charge l'expansion).

N

Navigateur application
Fenêtre spécialisée de l'application TwidoSoft qui affiche l'arborescence graphique d'une application. Facilite la configuration et l'affichage des applications.

O

Opérande
Nombre, repère ou symbole représentant une valeur qu'un programme peut manipuler dans une instruction.

Opérateur
Symbole ou code indiquant l'opération qu'une instruction doit réaliser.

P

PC Ordinateur personnel (de l'anglais Personal Computer).

PLS Génération d'impulsions. Bloc fonction qui génère une onde carrée avec un rapport cyclique de commutation de 50 %.

Point de réglage analogique Tension appliquée qui peut être réglée et convertie en une valeur numérique utilisable par une application.

Préférences Boîte de dialogue comprenant des options sélectionnables permettant de configurer les éditeurs de programmes par listes et par schémas de contacts.

Programmateur cyclique Bloc fonction dont le fonctionnement est semblable à celui des programmateurs cycliques électromécaniques qui permettent des modifications incrémentales (par pas) basées sur des événements externes.

Protection Se réfère aux deux types de protection d'une application : la protection par mot de passe, qui permet de contrôler l'accès à l'application, et la protection de l'application de l'automate, qui empêche toute personne non autorisée de visualiser ou de copier une application.

PWM Modulation de largeur d'impulsion (de l'anglais Pulse Width Modulation). Bloc fonction qui génère une onde carrée avec un rapport cyclique variable pouvant être défini par un programme.

R

RAM Mémoire vive (de l'anglais Random Access Memory). Les applications Twido sont téléchargées dans une mémoire RAM volatile interne afin d'être exécutées.

Redémarrage à chaud Après une coupure secteur, mise sous tension de l'automate sans modification de l'application. L'automate repasse à l'état dans lequel il était avant la coupure secteur et termine la scrutation qui était en cours. Toutes les données de l'application sont préservées. Cette fonction n'est disponible que sur les automates modulaires.

Références croisées Génération d'une liste d'opérandes, symboles, numéros de ligne/réseau et opérateurs utilisés dans une application pour simplifier la création et la gestion des applications.

Registres Registres spéciaux internes à l'automate dédiés aux blocs fonctions LIFO/FIFO.

Repères Registres internes de l'automate permettant de stocker les valeurs des variables, les constantes, les E/S, etc., du programme. Le symbole de pourcentage (%) utilisé en préfixe permet d'identifier les repères. Par exemple, %I0.1 indique un repère dans la mémoire RAM de l'automate contenant la valeur de la voie d'entrée 1.

Réseau Un réseau est un groupe d'éléments graphiques joints les uns aux autres par des liaisons horizontales ou verticales et placé entre deux barres de potentiel dans une grille. Un réseau peut comporter un maximum de sept lignes et onze colonnes.

Réseau schéma à contacts Affiche les parties d'un programme par listes qui ne sont pas réversibles en langage schéma à contacts.

RTC De l'anglais Real-Time Clock.Voir "Horodateur".

RTU De l'anglais Remote Terminal Unit. Protocole utilisant huit bits permettant d'échanger des données entre un automate et un PC.

S

Sauvegarder Commande permettant de copier l'application contenue dans la mémoire RAM de l'automate à la fois dans la mémoire EEPROM interne de l'automate et dans la cartouche de sauvegarde de mémoire en option (si elle est installée).

Scrutation Un automate scrute un programme et effectue principalement trois fonctions de base : Il lit d'abord les entrées et place les valeurs correspondantes dans la mémoire. Il exécute ensuite le programme d'application, instruction par instruction, puis stocke les résultats dans la mémoire. Il utilise ensuite les résultats pour mettre à jour les sorties.

Sortie réflexe En mode comptage, la valeur courante du compteur rapide (%VFC.V) est mesurée en fonction des seuils configurés afin de déterminer l'état des sorties dédiées.

Sorties seuil Bobines contrôlées directement par le compteur rapide (%VFC) en fonction des paramètres choisis lors de la configuration.

Symbole Un symbole est une chaîne de 32 caractères alphanumériques maximum, dont le premier caractère est alphabétique. Les symboles permettent de personnaliser les objets de l'automate afin de faciliter la maintenance de l'application.

Symboles non résolus Symbole sans repère de variable.

T

Table d'animation Table créée dans un éditeur de langage ou dans un écran de contrôle. Lorsqu'un PC est connecté à l'automate, une table d'animation permet de visualiser toutes les variables de l'automate et de forcer leurs valeurs lors d'un débogage. Elle peut être enregistrée dans un fichier distinct avec l'extension .tat.

Table de symboles Table des symboles utilisés dans une application. Affichée dans l'éditeur de symboles.

Temporisateur Bloc fonction utilisé pour sélectionner une plage de temps pour le contrôle d'un événement.

Twido Gamme d'automates Schneider Electric comprenant deux types d'automates (compacts et modulaires), des modules d'expansion pour ajouter des points d'E/S, et des options telles que l'horodateur, les communications, l'afficheur et les cartouches de sauvegarde de mémoire.

TwidoSoft Logiciel de développement graphique 32 bits fonctionnant sous Windows permettant de configurer et de programmer des automates Twido.

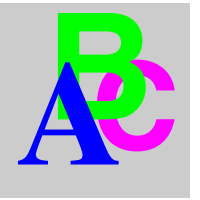
V

Validation auto par ligne Lors de l'insertion ou de la modification des instructions de liste, ce paramètre facultatif permet de valider les lignes de programme à mesure qu'elles sont saisies (recherche des erreurs et des symboles non résolus). Tous les éléments doivent être corrigés pour que le programmeur puisse quitter la ligne. Sélectionné à partir de la boîte de dialogue Préférences.

Variable Unité de mémoire pouvant être repérée et modifiée par un programme.

Variable de donnée Voir "Variable".

Index



Symbols

A

B

C

D

E

F

G

I

J

L

M

N

O

P

R

S

T

V

X

Z