SIEMENS

SIMATIC

Langage LOG pour SIMATIC S7-300/400

Manuel de référence

Ce manuel est livré avec la documentations référencée :
6ES7810-4CA10-8CW1

05/2010
A5E02790132-01

Mentions légales

Signalétique d'avertissement

Ce manuel donne des consignes que vous devez respecter pour votre propre sécurité et pour éviter des dommages matériels. Les avertissements servant à votre sécurité personnelle sont accompagnés d'un triangle de danger, les avertissements concernant uniquement des dommages matériels sont dépourvus de ce triangle. Les avertissements sont représentés ci-après par ordre décroissant de niveau de risque.

⚠DANGER
signifie que la non-application des mesures de sécurité appropriées **entraîne** la mort ou des blessures graves.

⚠ATTENTION
signifie que la non-application des mesures de sécurité appropriées **peut entraîner** la mort ou des blessures graves.

⚠PRUDENCE
accompagné d' un triangle de danger, signifie que la non-application des mesures de sécurité appropriées peut entraîner des blessures légères.

PRUDENCE
non accompagné d' un triangle de danger, signifie que la non-application des mesures de sécurité appropriées peut entraîner un dommage matériel.

IMPORTANT
signifie que le non-respect de l'avertissement correspondant peut entraîner l'apparition d'un événement ou d'un état indésirable.

En présence de plusieurs niveaux de risque, c'est toujours l'avertissement correspondant au niveau le plus élevé qui est reproduit. Si un avertissement avec triangle de danger prévient des risques de dommages corporels, le même avertissement peut aussi contenir un avis de mise en garde contre des dommages matériels.

Personnes qualifiées

L' appareil/le système décrit dans cette documentation ne doit être manipulé que par du **personnel qualifié** pour chaque tâche spécifique. La documentation relative à cette tâche doit être observée, en particulier les consignes de sécurité et avertissements. Les personnes qualifiées sont, en raison de leur formation et de leur expérience, en mesure de reconnaître les risques liés au maniement de ce produit / système et de les éviter.

Utilisation des produits Siemens conforme à leur destination

Tenez compte des points suivants:

⚠ATTENTION
Les produits Siemens ne doivent être utilisés que pour les cas d'application prévus dans le catalogue et dans la documentation technique correspondante. S'ils sont utilisés en liaison avec des produits et composants d'autres marques, ceux-ci doivent être recommandés ou agréés par Siemens. Le fonctionnement correct et sûr des produits suppose un transport, un entreposage, une mise en place, un montage, une mise en service, une utilisation et une maintenance dans les règles de l'art. Il faut respecter les conditions d'environnement admissibles ainsi que les indications dans les documentations afférentes.

Marques de fabrique

Toutes les désignations repérées par ® sont des marques déposées de Siemens AG. Les autres désignations dans ce document peuvent être des marques dont l'utilisation par des tiers à leurs propres fins peut enfreindre les droits de leurs propriétaires respectifs.

Exclusion de responsabilité

Nous avons vérifié la conformité du contenu du présent document avec le matériel et le logiciel qui y sont décrits. Ne pouvant toutefois exclure toute divergence, nous ne pouvons pas nous porter garants de la conformité intégrale. Si l'usage de ce manuel devait révéler des erreurs, nous en tiendrons compte et apporterons les corrections nécessaires dès la prochaine édition.

Siemens AG
Industry Sector
Postfach 48 48
90026 NÜRNBERG
ALLEMAGNE

A5E02790132-01
℗ 02/2010

Avant-propos

Objet du manuel

Ce manuel vous aidera à écrire des programmes utilisateur en langage LOG.

Il contient une partie de référence décrivant la syntaxe et le fonctionnement des éléments du langage de programmation LOG.

Connaissances fondamentales requises

Ce manuel s'adresse aux programmeurs souhaitant élaborer des programmes S7 ainsi qu'au personnel chargé de la mise en service et de la maintenance.

La compréhension du manuel requiert des connaissances générales dans le domaine de la technique d'automatisation.

Nous supposerons en outre des connaissances dans l'utilisation d'ordinateurs ou autres équipements (par exemple consoles de programmation) analogues au PC et des systèmes d'exploitation MS Windows XP, MS Windows Server 2003 ou MS Windows 7.

Domaine de validité du manuel

Le présent manuel est valable pour le logiciel STEP 7 V5.5.

Norme

LOG correspond au langage « Langage en blocs fonctionnels » défini dans la norme CEI 1131-3. Pour plus de renseignements à ce sujet, consultez la table de correspondance à la norme dans le fichier NORM_TBL.RTF (anglais) ou NORM_TAB.RTF (allemand) de STEP 7.

Connaissances requises

Vous trouverez dans l'aide en ligne de STEP 7 les connaissances thèoriques sur les programmes S7 nécessaires à la compréhension de ce manuel sur LOG. Les langages de programmation se basant sur le logiciel de base STEP 7, nous supposerons que vous savez utiliser ce logiciel et sa documentation.

Ce manuel fait partie de la documentation "STEP 7 Connaissances fondamentales“.

Le tableau suivant présente la documentation de STEP 7:

Manuel	**Objet**	**Numéro de référence**
STEP 7 Connaissances fondamentales avec • STEP 7 Getting Started • Programmer avec STEP 7 • Configuration matérielle et communication dans STEP 7 • STEP 7 Pour une transition facile de S5 à S7	Connaissances fondamentales pour le personnel technique. Décrit la marche à suivre pour réaliser des tâches d‘automatisation avec STEP 7 et S7-300/400.	6ES7810-4CA10-8CW0
STEP 7 Manuels de référence sur les • Langages CONT/LOG/LIST pour SIMATIC S7-300/400 • Logiciel système pour SIMATIC S7-300/400 Fonctions standard et fonctions système Volume 1 et Volume 2	Manuels de référence décrivant les langages de programmation CONT, LOG et LIST de même que les fonctions standard et les fonctions système en complément des connaissances fondamentales de STEP 7.	6ES7810-4CA10-8CW1

Aides en ligne	**Objet**	**Numéro de référence**
Aide de STEP 7	Connaissances fondamentales pour la programmation ainsi que pour la configuration du matériel avec STEP 7, sous forme d‘aide en ligne.	Fait partie du logiciel STEP 7
Aides de référence de LIST/CONT/LOG Aide de référence sur les SFB/SFC Aide de référence sur les blocs d‘organisation	Aides en ligne contextuelles de référence	Fait partie du logiciel STEP 7

Aide en ligne

En complément au manuel, l'aide en ligne intégrée au logiciel vous offre une assistance détaillée lors de l'utilisation du logiciel.

Ce système d'aide est intégré au logiciel grâce à plusieurs interfaces :

- L'aide contextuelle donne des informations sur le contexte actuel, par exemple sur une boîte de dialogue ouverte ou sur une fenêtre active. Vous l'appelez en cliquant sur le bouton "Aide" ou en appuyant sur la touche F1.
- Le menu d'aide **?** propose plusieurs commandes : **Rubrique d'aides** ouvre le sommaire de l'aide de STEP 7.
- Vous obtenez le glossaire relatif à toutes les applications de STEP 7 en cliquant sur **"Glossaire".**

Ce manuel est extrait de l' "Aide pour LOG". En raison de la structure similaire entre le manuel et l'aide en ligne, le passage de l'un à l'autre est aisé.

Assistance supplémentaire

Si des questions sont restées sans réponse dans ce manuel, veuillez vous adresser à votre interlocuteur Siemens dans la filiale ou l'agence de votre région.

Vous trouvez votre interloculeur sous :

http://www.siemens.com/automation/partner

Vous trouvez un fil rouge pour la recherche de documentations techniques sur les produits et systèmes SIMATIC à l'adresse suivante sur Internet :

http://www.siemens.com/simatic-tech-doku-portal

Le catalogue en ligne et le système de commande en ligne se trouvent à l'adresse :

http://mall.automation.siemens.com/

Centre de formation SIMATIC

Nous proposons des cours de formation pour vous faciliter l'apprentissage des automates programmables SIMATIC S7. Veuillez vous adresser à votre centre de formation régional ou au centre principal à D 90026 Nuremberg.

Internet: http://www.sitrain.com

Technical Support

Vous pouvez joindre le support technique pour tous les produits d'Industry Automation.

- Via le formulaire Web de demande d'assistance (Support Request)
  http://www.siemens.com/automation/support-request

Vous trouvez plus d'informations concernant notre Technical Support sur Internet à l'adresse suivante :

http://www.siemens.com/automation/service

Service & Support sur Internet

En plus de la documentation offerte, vous trouvez la totalité de notre savoir-faire en ligne sur Internet à l'adresse suivante :

http://www.siemens.com/automation/service&support

Vous y trouvez :

- le bulletin d'informations qui vous fournit constamment les dernières informations sur le produit,
- les documents dont vous avez besoin à l'aide de la fonction de recherche du Support produit,
- le forum où utilisateurs et spécialistes peuvent échanger des informations,
- votre interlocuteur Industry Automation sur site,
- des informations sur les réparations, pièces de rechange et la consultation.

Sommaire

1 Opérations combinatoires sur bits

1.1 Vue d'ensemble des opérations combinatoires sur bits

Description

Les opérations combinatoires sur bits utilisent deux chiffres : 1 et 0. Ces deux chiffres sont à la base du système de numération binaire et sont appelés chiffres binaires ou bits. Le 1 signifie « OUI logique » et le 0 « NON logique » en relation avec ET, OU, OU exclusif et des sorties.

Les opérations de combinaison sur bits évaluent les états de signal 1 et 0 et les combinent selon la logique booléenne. Le résultat de ces combinaisons est égal à 1 ou 0. Il s'agit du résultat logique (RLG).

Il existe des opérations combinatoires sur bits pour effectuer les fonctions suivantes :

- ET, OU et Exclusiv-OU interrogent chacune l'état de signal et leur résultat est soit copié dans le bit de résultat logique RLG, soit combiné au RLG.
- Combinaisons ET avant OU et OU avant ET
- Affectation et Connecteur assignent le RLG ou le mémorisent temporairement.

Les opérations suivantes réagissent à un RLG égal à 1 :

- S : Mettre à 1
- R : Mettre à 0
- SR : Bascule mise à 1, mise à 0
- RS : Bascule mise à 0, mise à 1

D'autres opérations exécutent les fonctions suivantes en cas de front montant ou descendant :

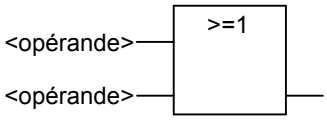
- N : Détecter front descendant
- P : Détecter front montant
- NEG : Détecter front descendant de signal
- POS : Détecter front montant de signal

Les opérations suivantes agissent directement sur le RLG de la manière suivante :

- Insérer entrée binaire
- Inverser entrée binaire
- SAVE : Sauvegarder RLG dans RB

1.2 >=1 : Porte OU

Représentation



Paramètre	Type de données	Zone de mémoire	Description
<opérande>	BOOL	E, A, M, T, Z, D, L	L'opérande indique le bit dont l'état de signal est interrogé.

Description

La porte **OU** vous permet d'interroger l'état de signal de deux opérandes ou plus indiqués aux entrées d'une boîte OU.

Si l'état de signal d'un des opérandes est 1, la condition est réalisée et l'opération fournit un résultat égal à 1. Si tous les opérandes ont l'état de signal 0, la condition n'est pas satisfaite et l'opération fournit un résultat égal à 0.

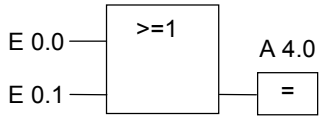
Lorsque la porte **OU** est la première opération dans une séquence combinatoire, elle range le résultat de son interrogation d'état de signal dans le bit de résultat logique (RLG).

Lorsqu'elle n'est pas la première opération dans la séquence combinatoire, elle combine le résultat de son interrogation d'état de signal à la valeur figurant dans le bit RLG. Cette combinaison se fait selon la table de vérité OU.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	-	-	-	-	X	X	X	1

Exemple

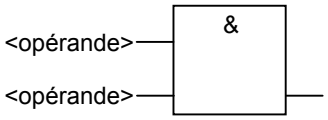


La sortie A 4.0 est mise à 1

- si l'état de signal est 1 à l'entrée E 0.0 **ou** à l'entrée E 0.1.

1.3 & : Porte ET

Représentation



Paramètre	Type de données	Zone de mémoire	Description
<opérande>	BOOL	E, A, M, T, Z, D, L	L'opérande indique le bit dont l'état de signal est interrogé.

Description

La porte **ET** vous permet d'interroger l'état de signal de deux opérandes ou plus indiqués aux entrées d'une boîte ET.

Si l'état de signal de tous les opérandes est 1, la condition est réalisée et l'opération fournit un résultat égal à 1. En revanche, si un des opérandes a l'état de signal 0, la condition n'est pas satisfaite et l'opération fournit un résultat égal à 0.

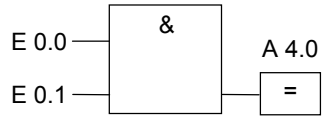
Lorsque la porte **ET** est la première opération dans une séquence combinatoire, elle range le résultat de son interrogation d'état de signal dans le bit de résultat logique (RLG).

Lorsqu'elle n'est pas la première opération dans la séquence combinatoire, elle combine le résultat de son interrogation d'état de signal à la valeur figurant dans le bit RLG. Cette combinaison se fait selon la table de vérité ET.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	-	-	-	-	X	X	X	1

Exemple



La sortie A 4.0 est mise à 1

si l'état de signal est 1 aux entrées E 0.0 **et** E 0.1.

1.4 Combinaisons ET avant OU et OU avant ET

Description

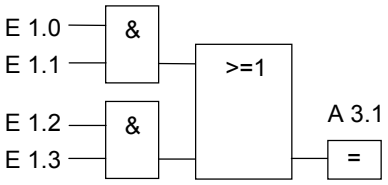
L'opération **ET avant OU** vous permet d'interroger le résultat d'une interrogation d'état de signal selon la table de vérité OU.

Pour une combinaison **ET avant OU**, l'état de signal est 1 lorsqu'une combinaison ET au moins est satisfaite.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	-	-	-	-	X	X	X	1

Exemple



L'état de signal est 1 à la sortie A 3.1 si une combinaison ET au moins est satisfaite.

L'état de signal est 0 à la sortie A 3.1 si aucune combinaison ET n'est satisfaite.

Description

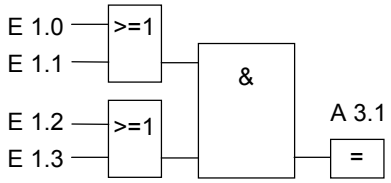
L'opération **OU avant ET** vous permet d'interroger le résultat d'une interrogation d'état de signal selon la table de vérité ET.

Pour une combinaison **OU avant ET**, l'état de signal est 1 lorsque toutes les combinaisons OU sont satisfaites.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	-	-	-	-	X	X	X	1

Exemple

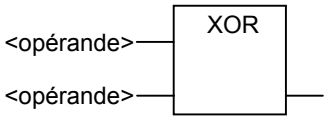


L'état de signal est 1 à la sortie A 3.1 si les deux combinaisons OU sont satisfaites.

L'état de signal est 0 à la sortie A 3.1 si une combinaison OU au moins n'est pas satisfaite.

1.5 XOR : Combinaison OU exclusif

Représentation



Paramètre	Type de données	Zone de mémoire	Description
<opérande>	BOOL	E, A, M, T, Z, D, L	L'opérande indique le bit dont l'état de signal est interrogé.

Description

L'opération **OU exclusif** vous permet d'interroger le résultat d'une interrogation d'état de signal selon la table de vérité OU exclusif.

Pour une combinaison **OU exclusif**, l'état de signal est 1 lorsque l'état de signal de l'un des deux opérandes est 1.

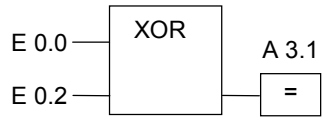
Si la fonction **XOR** comprend plus de 2 opérandes, la règle est la suivante :

Si un nombre impair d'opérandes interrogés fournit en résultat "1", le résultat logique de ces opérandes est "1".

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	-	-	-	-	X	X	X	1

Exemple



L'état de signal est 1 à la sortie A 3.1 lorsque **soit** l'état de signal est 1 à l'entrée E 0.0 **soit** il est 1 à l'entrée E 0.2, et ce exclusivement.

1.6 Insérer entrée binaire

Représentation

<opérande>

Paramètre	Type de données	Zone de mémoire	Description
<opérande>	BOOL	E, A, M, T, Z, D, L	L'opérande indique le bit dont l'état de signal est interrogé.

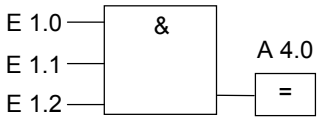
Description

L'opération **Insérer entrée binaire** insère une nouvelle entrée binaire après l'entrée sélectionnée d'une boîte ET, OU ou OU exclusif.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	-	-	-	-	-	1	X	-

Exemple



La sortie A 4.0 est à 1 si l'état de signal est 1 aux entrées E 1.0 **et** E 1.1 **et** E 1.2.

1.7 Inverser l'entrée binaire

Représentation

Description

L'opération **Inverser entrée binaire** inverse le résultat logique.

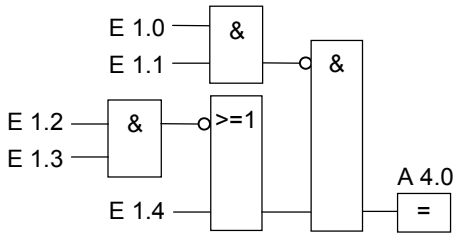
Vous devez tenir compte de certaines règles lors de la négation du résultat logique :

- Il n'y a pas de mise entre parenthèses si le résultat logique est inversé au niveau de la première entrée d'une porte ET ou d'une porte OU.
- Si le résultat logique n'est pas inversé au niveau de la première entrée d'une porte OU, toute la combinaison binaire avant l'entrée est incluse dans la combinaison OU.
- Si le résultat logique n'est pas inversé au niveau de la première entrée d'une porte ET, toute la combinaison binaire avant l'entrée est incluse dans la combinaison ET.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	-	-	-	-	-	1	X	-

Exemple

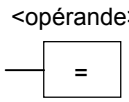


La sortie A 4.0 est à 1 :

- si l'état de signal **n'est pas** 1 aux entrées E 1.0 **et** E 1.1
- **et** que l'état de signal **n'est pas** 1 aux entrées E 1.2 **et** E 1.3
- **ou** que l'état de signal **n'est pas** 1 à l'entrée E 1.4.

1.8 = : Affectation

Représentation



Paramètre	Type de données	Zone de mémoire	Description
<opérande>	BOOL	E, A, M, D, L	L'opérande indique le bit auquel l'état de signal de la séquence combinatoire est affecté.

Description

L'opération **Affectation** fournit le résultat logique. La boîte au bout de la combinaison délivre le signal 1 ou 0 selon les critères suivants :

- La sortie prend le signal 1 lorsque les conditions de la combinaison avant la boîte de sortie sont satisfaites.
- Elle prend le signal 0 lorsque les conditions de la combinaison avant la boîte de sortie ne sont pas satisfaites.

La combinaison LOG affecte l'état de signal à la sortie à laquelle accède l'opération (il reviendrait au même d'affecter l'état de signal du bit RLG à l'opérande). Si les conditions des combinaisons LOG sont satisfaites, l'état de signal est 1 à la boîte de sortie ; sinon, il est égal à 0.

Le relais de masquage MCR influence l'opération **Affectation**.

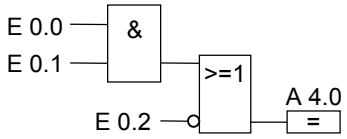
Vous ne pouvez placer la boîte "Affectation" qu'à l'extrémité droite de la séquence combinatoire. Il est toutefois possible d'en utiliser plusieurs.

Vous pouvez créer une affectation inversée à l'aide de l'opération **Inverser entrée binaire**.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	-	-	-	-	0	X	-	0

Exemple

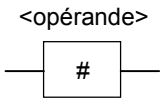


L'état de signal de la sortie A 4.0 est 1 :

- si l'état de signal est 1 aux entrées E 0.0 **et** E 0.1
- **ou** si l'état de signal est 0 à l'entrée E 0.2

1.9 # : Connecteur

Représentation



Paramètre	Type de données	Zone de mémoire	Description
<opérande>	BOOL	E, A, M, D, *L	L'opérande indique le bit auquel le résultat logique est affecté.

* Dans l'opération **Connecteur**, vous ne pouvez utiliser un opérande dans la zone de mémoire L que si vous le déclarez la table de déclaration de variables.

Description

L'opération **Connecteur** est un élément d'affectation intermédiaire qui mémorise le RLG. Cet élément sauvegarde la combinaison sur bits de la dernière branche ouverte avant l'élément d'affectation.

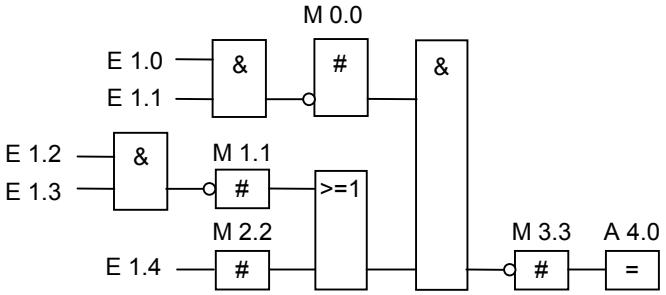
Le relais de masquage MCR influence l'opération **Connecteur**.

Vous pouvez créer un connecteur inversé en inversant l'entrée du connecteur.

Mot d'état

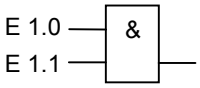
	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	-	-	-	-	0	X	-	1

Exemple

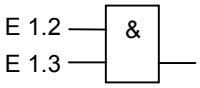


Les connecteurs mémorisent les résultats de combinaison suivants :

M 0.0 mémorise le RLG inversé de



M 1.1 mémorise le RLG inversé de

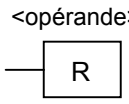


M 2.2 mémorise le RLG de E 1.4

et M 3.3 mémorise le RLG inversé de la combinaison sur bits.

1.10 R : Mettre à 0

Représentation



Paramètre	Type de données	Zone de mémoire	Description
<opérande>	BOOL TIMER COUNTER	E, A, M, T, Z, D, L	L'opérande indique le bit qui doit être mis à 0.

Description

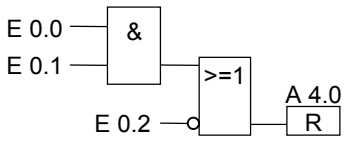
L'opération **Mettre à 0** ne s'exécute que si le RLG a la valeur 1. Dans ce cas, l'opération met son opérande à 0. Si le RLG égale 0, l'opération n'a pas d'effet sur l'opérande précisé qui reste inchangé.

Le relais de masquage MCR influence l'opération **Mettre à 0**.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	-	-	-	-	0	X	-	0

Exemple



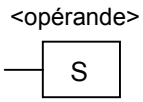
L'état de signal de la sortie A 4.0 est mis à 0 si :

- l'état de signal est 1 aux entrées E 0.0 **et** E 0.1
- **ou** l'état de signal est 0 à l'entrée E 0.2.

Si le RLG de la branche est égal à 0, l'état de signal de la sortie A 4.0 reste inchangé.

1.11 S : Mettre à 1

Représentation



Paramètre	Type de données	Zone de mémoire	Description
<opérande>	BOOL	E, A, M, D, L	L'opérande indique le bit qui doit être mis à 1.

Description

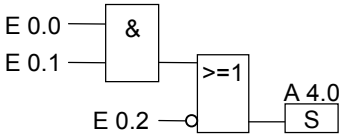
L'opération **Mettre à 1** ne s'exécute que si le RLG a la valeur 1. Dans ce cas, l'opération met son opérande à 1. Si le RLG égale 0, l'opération n'a pas d'effet sur l'opérande précisé qui reste inchangé.

Le relais de masquage MCR influence l'opération **Mettre à 1**.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	-	-	-	-	0	X	-	0

Exemple



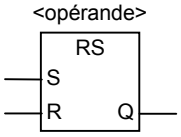
L'état de signal de la sortie A 4.0 est mis à 1 :

- si l'état de signal est 1 aux entrées E 0.0 **et** E 0.1
- **ou** si l'état de signal est 0 à l'entrée E 0.2.

Si le RLG de la branche est égal à 0, l'état de signal de la sortie A 4.0 reste inchangé.

1.12 RS : Bascule mise à 0, mise à 1

Représentation



Paramètre	Type de données	Zone de mémoire	Description
<opérande>	BOOL	E, A, M, D, L	L'opérande indique le bit qui doit être mis à 1 ou à 0.
S	BOOL	E, A, M, D, L, T, Z	Mise à 0 activée
R	BOOL	E, A, M, D, L, T, Z	Mise à 1 activée
Q	BOOL	E, A, M, D, L	Etat de signal de l'<opérande>

Description

L'opération **Bascule mise à 0, mise à 1** n'exécute la mise à 1 (S) et la mise à 0 (R) que lorsque le RLG est égal à 1. Lorsque le RLG est égal à 0, l'opérande précisé dans l'opération reste inchangé.

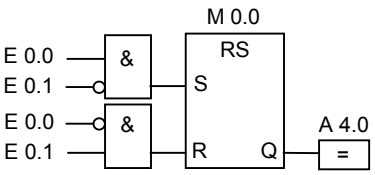
Une **Bascule mise à 0, mise à 1** est mise à 1 si l'état de signal est 1 à l'entrée R et 0 à l'entrée S. Si l'état de signal est 0 à l'entrée R et 1 à l'entrée S, la bascule est mise à 1. Si le RLG est égal à 1 aux deux entrées, la bascule est mise à 1.

Le relais de masquage MCR influence l'opération **Bascule mise à 0, mise à 1**.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	-	-	-	-	X	X	X	1

Exemple

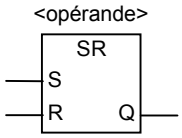


Si l'état de signal est 1 à l'entrée E 0.0 et 0 à l'entrée E 0.1, le mémento M 0.0 est mis à 0 et la sortie A 4.0 est à 0. Si l'état de signal est 0 à l'entrée E 0.0 et 1 à l'entrée E 0.1, le mémento M 0.0 est mis à 1 et la sortie A 4.0 est à 1.

Si les deux états de signal ont la valeur 0, rien ne se passe. En revanche, s'ils ont tous deux la valeur 1, la mise à 1, exécutée en dernier, l'emporte : M 0.0 est mis à 1 et la sortie A 4.0 est à 1.

1.13 SR : Bascule mise à 1, mise à 0

Représentation



Paramètre	Type de données	Zone de mémoire	Description
<opérande>	BOOL	E, A, M, D, L	L'opérande indique le bit qui doit être mis à 1 ou à 0.
S	BOOL	E, A, M, D, L, T, Z	Mise à 1 activée
R	BOOL	E, A, M, D, L, T, Z	Mise à 0 activée
Q	BOOL	E, A, M, D, L	Etat de signal de l'<opérande>

Description

L'opération **Bascule mise à 1, mise à 0** n'exécute la mise à 1 (S) et la mise à 0 (R) que lorsque le RLG est égal à 1. Lorsque le RLG est égal à 0, l'opérande précisé dans l'opération reste inchangé.

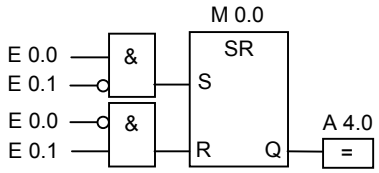
Une **Bascule mise à 1, mise à 0** est mise à 1 si l'état de signal est 1 à l'entrée S et 0 à l'entrée R. Si l'état de signal est 0 à l'entrée S et 1 à l'entrée R, la bascule est mise à 0. Si le RLG est égal à 1 aux deux entrées, la bascule est mise à 0.

Le relais de masquage MCR influence l'opération **Bascule mise à 1, mise à 0**.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	-	-	-	-	X	X	X	1

Exemple

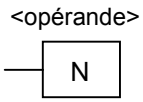


Si l'état de signal est 1 à l'entrée E 0.0 et 0 à l'entrée E 0.1, le mémento M 0.0 est mis à 1 et la sortie A 4.0 est à 1. Si l'état de signal est 0 à l'entrée E 0.0 et 1 à l'entrée E 0.1, le mémento M 0.0 est mis à 0 et la sortie A 4.0 est à 0.

Si les deux états de signal ont la valeur 0, rien ne se passe. En revanche, s'ils ont tous deux la valeur 1, la mise à 0, exécutée en dernier, l'emporte : M 0.0 est mis à 0 et la sortie A 4.0 est à 0.

1.14 N : Détecter front descendant

Représentation



Paramètre	Type de données	Zone de mémoire	Description
<opérande>	BOOL	E, A, M, D, L	L'opérande indique le mémento de front qui mémorise l'ancien RLG.

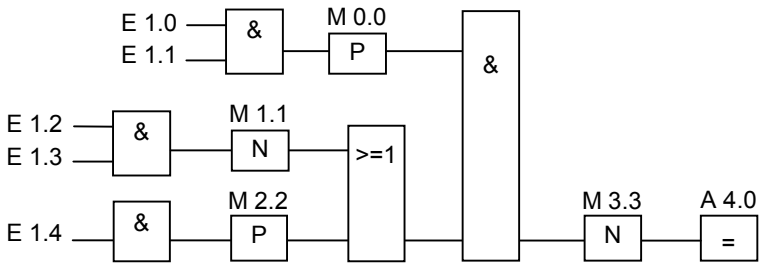
Description

L'opération **Détecter front descendant** détecte le passage de 1 à 0 dans l'opérande indiqué (front descendant) et montre cette transition avec un RLG égal à 1 après cette opération. L'état de signal en cours du RLG est comparé à celui de l'opérande (mémento de front). Si l'état de signal de l'opérande est 1 et le RLG avant l'opération est 0, le RLG est à 1 après l'opération (impulsion) ; dans tous les autres cas, le RLG est à 0. Le RLG avant l'opération est sauvegardé dans l'opérande.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	-	-	-	-	0	X	X	1

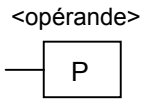
Exemple



Le mémento de front M 3.3 mémorise l'état de signal du RLG précédent.

1.15 P : Détecter front montant

Représentation



Paramètre	Type de données	Zone de mémoire	Description
<opérande>	BOOL	E, A, M, D, L	L'opérande indique le mémento de front qui mémorise l'ancien RLG.

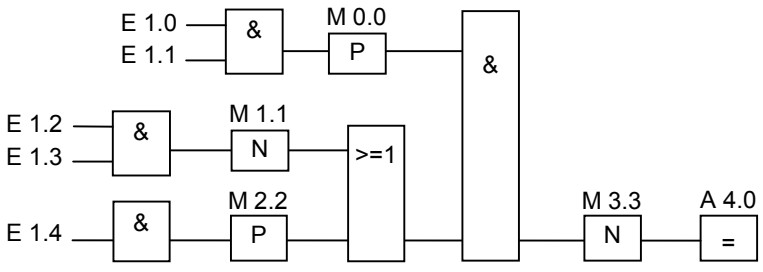
Description

L'opération **Détecter front montant** détecte le passage de 0 à 1 dans l'opérande indiqué (front montant) et montre cette transition avec un RLG égal à 1 après cette opération. L'état de signal en cours du RLG est comparé à celui de l'opérande (mémento de front). Si l'état de signal de l'opérande est 0 et le RLG avant l'opération est 1, le RLG est à 1 après l'opération (impulsion) ; dans tous les autres cas, le RLG est à 0. Le RLG avant l'opération est sauvegardé dans l'opérande.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	-	-	-	-	0	X	X	1

Exemple



Le mémento de front M 0.0 mémorise l'état de signal du RLG précédent.

1.16 SAVE : Sauvegarder RLG dans RB

Représentation

Description

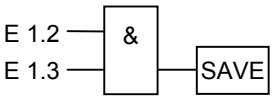
L'opération **Sauvegarder RLG dans RB** sauvegarde le résultat logique dans le bit de résultat binaire du mot d'état. Elle ne remet toutefois pas à 1 le bit de première interrogation /PI, ce qui entraîne, en cas de combinaison ET dans le réseau suivant, la prise en compte de l'état du bit RB.

Contrairement à ce que vous trouvez spécifié dans le manuel, il convient d'utiliser l'opération "SAVE" (CONT, LOG, LIST) dans les cas suivants :
L'utilisation de SAVE suivie d'une interrogation du bit RB dans le même bloc ou dans des blocs subordonnés n'est pas recommandée car le bit RB risque d'être modifié plusieurs fois durant les nombreuses opérations exécutées entre. Utilisez plutôt l'opération **SAVE** avant de quitter un bloc, car la sortie de validation ENO (= bit RB) est mise à la valeur du bit RLG et que vous pouvez enchaîner par le dépistage d'erreurs du bloc.

Mot d'état

	RB	**BI1**	**BI0**	**DEB**	**DM**	**OU**	**ETAT**	**RLG**	**/PI**
Ecriture	X	-	-	-	-	-	-	-	-

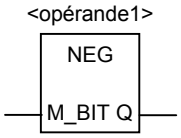
Exemple



Le résultat logique est sauvegardé dans le bit de résultat binaire.

1.17 NEG : Détecter front descendant de signal

Représentation



Paramètre	Type de données	Zone de mémoire	Description
<opérande1>	BOOL	E, A, M, D, L	Signal à interroger pour détecter un front descendant
M_BIT	BOOL	A, M, D	L'opérande M_BIT indique le mémento de front qui mémorise l'état de signal précédent de NEG. N'utilisez la mémoire image des entrées (E) pour M_BIT que si aucun module d'entrées n'occupe déjà cet opérande.
Q	BOOL	E, A, M, D, L	Sortie monostable

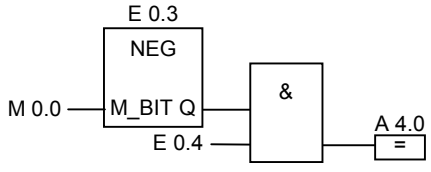
Description

L'opération **Détecter front descendant de signal** compare l'état de signal de <opérande1> à celui provenant de l'interrogation précédente figurant dans le paramètre M_BIT. En cas de passage de 1 à 0, la sortie Q est mise à 1 ; dans tous les autres cas, elle est mise à 0.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	-	-	-	-	0	1	X	1

Exemple

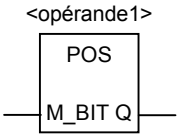


La sortie A 4.0 est à 1 si :

- il y a un front descendant à l'entrée E 0.3
- **et** l'état de signal est 1 à l'entrée E 0.4.

1.18 POS : Détecter front montant de signal

Représentation



Paramètre	Type de données	Zone de mémoire	Description
<opérande1>	BOOL	E, A, M, D, L	Signal à interroger pour détecter un front montant
M_BIT	BOOL	A, M, D	L'opérande M_BIT indique le mémento de front qui mémorise l'état de signal précédent de POS. N'utilisez la mémoire image des entrées (E) pour M_BIT que si aucun module d'entrées n'occupe déjà cet opérande.
Q	BOOL	E, A, M, D, L	Sortie monostable

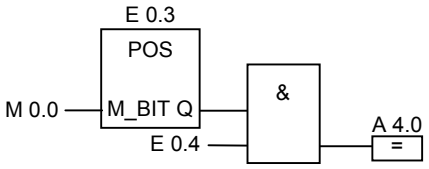
Description

L'opération **Détecter front montant de signal** compare l'état de signal de <opérande1> à celui provenant de l'interrogation précédente figurant dans le paramètre M_BIT. En cas de passage de 0 à 1, la sortie Q est mise à 1 ; dans tous les autres cas, elle est mise à 0.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	-	-	-	-	0	1	X	1

Exemple



La sortie A 4.0 est à 1 si :

- il y a un front montant à l'entrée E 0.3
- **et** l'état de signal est 1 à l'entrée E 0.4.

2 Opérations de comparaison

2.1 Vue d'ensemble des opérations de comparaison

Description

Les opérations de comparaison comparent les entrées IN1 et IN2 selon les types de comparaison suivants :

==	IN1 égal à IN2
<>	IN1 différent de IN2
>	IN1 supérieur à IN2
<	IN1 inférieur à IN2
>=	IN1 supérieur ou égal à IN2
<=	IN1 inférieur ou égal à IN2

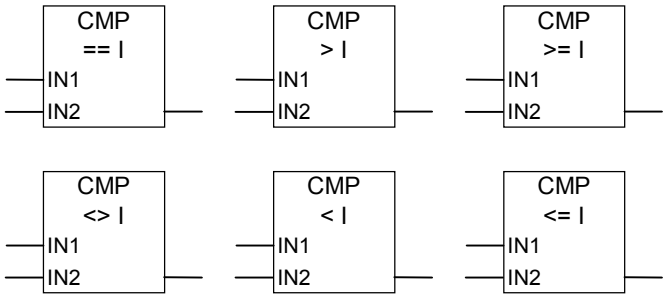
Si la comparaison est vraie, son résultat logique (RLG) est 1 ; sinon, il est égal à 0. Il n'y a pas de négation du résultat de la comparaison, car cela peut être obtenu à l'aide de l'opération de comparaison inverse.

Vous disposez des opérations de comparaison suivantes :

- CMP ? I : Comparer entiers de 16 bits
- CMP ? D : Comparer entiers de 32 bits
- CMP ? R : Comparer nombres réels

2.2 CMP ? I : Comparer entiers de 16 bits

Représentation



Paramètre	Type de données	Zone de mémoire	Description
IN1	INT	E, A, M, D, L ou constante	Premier terme de la comparaison
IN2	INT	E, A, M, D, L ou constante	Second terme de la comparaison
Sortie boîte	BOOL	E, A, M, D, L	Résultat de la comparaison

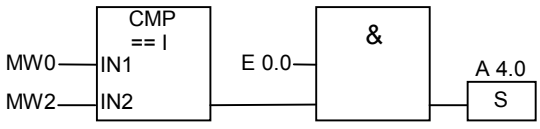
Description

L'opération **Comparer entiers de 16 bits** compare les entrées IN1 et IN2 - deux nombres entiers de 16 bits - selon le type de comparaison que vous avez sélectionné.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	X	X	X	0	-	0	X	X	1

Exemple

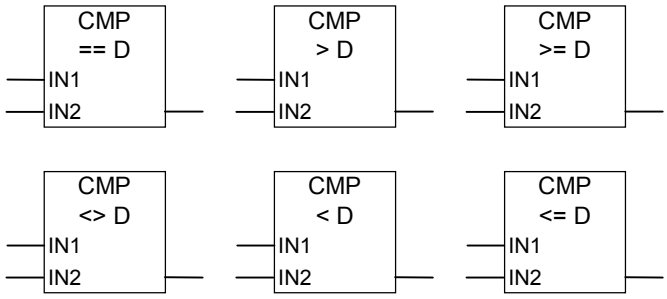


La sortie A 4.0 est mise à 1 si :

- MW0 = MW2
- **et** l'état de signal est 1 à l'entrée E 0.0.

2.3 CMP ? D : Comparer entiers de 32 bits

Représentation



Paramètre	Type de données	Zone de mémoire	Description
IN1	DINT	E, A, M, D, L ou constante	Premier terme de la comparaison
IN2	DINT	E, A, M, D, L ou constante	Second terme de la comparaison
Sortie boîte	BOOL	E, A, M, D, L	Résultat de la comparaison

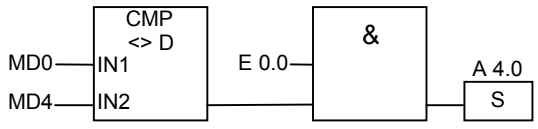
Description

L'opération **Comparer entiers de 32 bits** compare les entrées IN1 et IN2 - deux nombres entiers de 32 bits - selon le type de comparaison que vous avez sélectionné.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	X	X	0	-	0	X	X	1

Exemple

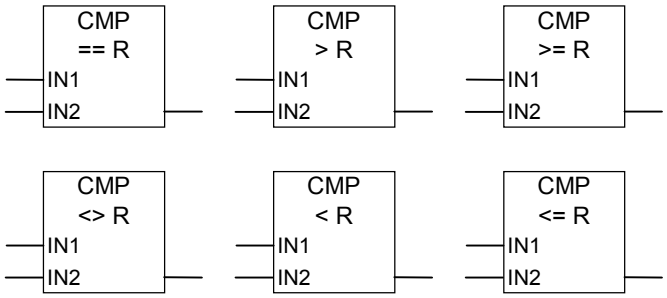


La sortie A 4.0 est mise à 1 si :

- MD0 est différent de MD4
- **et** l'état de signal est 1 à l'entrée E 0.0.

2.4 CMP ? R : Comparer nombres réels

Représentation



Paramètre	Type de données	Zone de mémoire	Description
IN1	REAL	E, A, M, D, L ou constante	Premier terme de la comparaison
IN2	REAL	E, A, M, D, L ou constante	Second terme de la comparaison
Sortie boîte	BOOL	E, A, M, D, L	Résultat de la comparaison

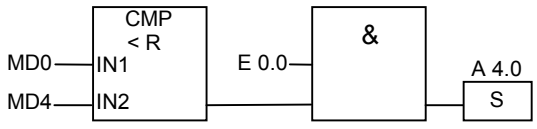
Description

L'opération **Comparer nombres réels** compare les entrées IN1 et IN2 - deux nombres réels - selon le type de comparaison que vous avez sélectionné.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	X	X	X	X	0	X	X	1

Exemple



La sortie A 4.0 est mise à 1 si :

- MD0 < MD4
- **et** l'état de signal est 1 à l'entrée E 0.0.

3 Opérations de conversion

3.1 Vue d'ensemble des opérations de conversion

Description

Les opérations suivantes permettent de convertir des nombres décimaux codés binaires et des nombres entiers en d'autres types de nombres :

- BCD_I : Convertir nombre DCB en entier de 16 bits
- I_BCD : Convertir entier de 16 bits en nombre DCB
- BCD_DI : Convertir nombre DCB en entier de 32 bits
- I_DI : Convertir entier de 16 bits en entier de 32 bits
- DI_BCD : Convertir entier de 32 bits en nombre DCB
- DI_R : Convertir entier de 32 bits en nombre réel

Les opérations suivantes permettent de former le complément de nombres entiers ou de réaliser l'inversion de nombres à virgule flottante :

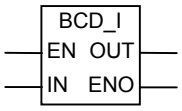
- INV_I : Complément à 1 d'entier de 16 bits
- INV_DI : Complément à 1 d'entier de 32 bits
- NEG_I : Complément à 2 d'entier de 16 bits
- NEG_DI : Complément à 2 d'entier de 32 bits
- NEG_R : Inverser le signe d'un nombre réel

Les opérations suivantes permettent de convertir le nombre à virgule flottante IEEE de 32 en un nombre entier de 32 bits (entier double). Les différentes opérations se distinguent par leur façon d'arrondir :

- ROUND : Arrondir à entier de 32 bits
- TRUNC : Tronquer à la partie entière (32 bits)
- CEIL : Convertir nombre réel en entier supérieur le plus proche
- FLOOR : Convertir nombre réel en entier inférieur le plus proche

3.2 BCD_I : Convertir nombre DCB en entier de 16 bits

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN	WORD	E, A, M, D, L ou constante	Nombre en format DCB
OUT	INT	E, A, M, D, L	Valeur entière de 16 bits du nombre DCB
ENO	BOOL	E, A, M, D, L	Sortie de validation

Description

L'opération **Convertir nombre DCB en entier de 16 bits** lit le contenu du paramètre d'entrée IN comme nombre décimal codé binaire à trois chiffres (DCB, ± 999), le convertit en un nombre entier de 16 bits et range le résultat dans le paramètre de sortie OUT.

ENO et EN ont toujours un état de signal identique.

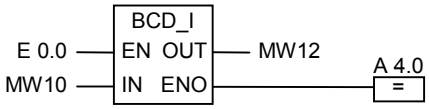
Si l'un des chiffres du nombre DCB se trouve dans la plage non autorisée entre 10 et 15, une erreur DCB se produit durant la conversion :

- La CPU passe à l'état de fonctionnement "Arrêt" (STOP). La mémoire tampon de diagnostic indique une "Erreur de conversion DCB" dont le numéro d'événement est 2521.
- S'il a été programmé, l'OB121 est appelé.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	1	-	-	-	-	0	1	1	1

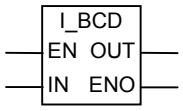
Exemple



La conversion est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le contenu du mot de mémento MW10 est lu comme nombre DCB à trois chiffres et converti en nombre entier de 16 bits. Le résultat est rangé dans MW12. La sortie A 4.0 est mise à 1 si la conversion est exécutée (ENO = EN).

3.3 I_BCD : Convertir entier de 16 bits en nombre DCB

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN	INT	E, A, M, D, L ou constante	Nombre entier de 16 bits
OUT	WORD	E, A, M, D, L	Valeur DCB du nombre entier de 16 bits
ENO	BOOL	E, A, M, D, L	Sortie de validation

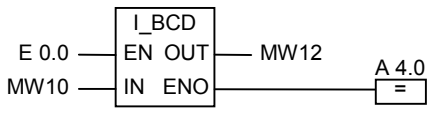
Description

L'opération **Convertir entier de 16 bits en nombre DCB** lit le contenu du paramètre d'entrée IN comme nombre entier de 16 bits, le convertit en un nombre décimal codé binaire à trois chiffres (DCB, ± 999) et range le résultat dans le paramètre de sortie OUT. En cas de débordement, ENO est mis à 0.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	X	-	-	X	X	0	X	X	1

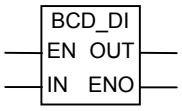
Exemple



La conversion est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le contenu du mot de mémento MW10 est lu comme nombre entier de 16 bits et converti en nombre DCB à trois chiffres. Le résultat est rangé dans MW12. En cas de débordement, la sortie A 4.0 est mise à 0. Si l'état de signal à l'entrée EN est égal à 0 (la conversion n'est pas exécutée), l'état de signal à la sortie A 4.0 est également 0.

3.4 BCD_DI : Convertir nombre DCB en entier de 32 bits

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN	DWORD	E, A, M, D, L ou constante	Nombre en format DCB
OUT	DINT	E, A, M, D, L	Valeur entière de 32 bits du nombre DCB
ENO	BOOL	E, A, M, D, L	Sortie de validation

Description

L'opération **Convertir nombre DCB en entier de 32 bits** lit le contenu du paramètre d'entrée IN comme nombre décimal codé binaire à sept chiffres (DCB, $\pm$ 9 999 999), le convertit en un nombre entier de 32 bits et range le résultat dans le paramètre de sortie OUT.

ENO et EN ont toujours un état de signal identique.

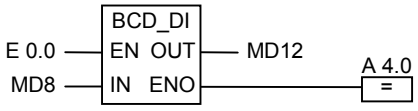
Si l'un des chiffres du nombre DCB se trouve dans la plage non autorisée entre 10 et 15, une erreur DCB se produit durant la conversion :

- La CPU passe à l'état de fonctionnement "Arrêt" (STOP). La mémoire tampon de diagnostic indique une "Erreur de conversion DCB" dont le numéro d'événement est 2521.
- S'il a été programmé, l'OB121 est appelé.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	1	-	-	-	-	0	1	1	1

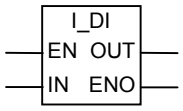
Exemple



La conversion est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le contenu du double mot de mémento MD8 est lu comme nombre DCB à sept chiffres et converti en nombre entier de 32 bits. Le résultat est rangé dans MD12. La sortie A 4.0 est mise à 1 si la conversion est exécutée (ENO = EN).

3.5 I_DI : Convertir entier de 16 bits en entier de 32 bits

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN	INT	E, A, M, D, L ou constante	Valeur à convertir
OUT	DINT	E, A, M, D, L	Résultat
ENO	BOOL	E, A, M, D, L	Sortie de validation

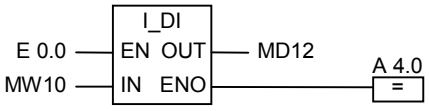
Description

L'opération **Convertir entier de 16 bits en entier de 32 bits** lit le contenu du paramètre d'entrée IN comme nombre entier de 16 bits, le convertit en un nombre entier de 32 bits et range le résultat dans le paramètre de sortie OUT. ENO et EN ont toujours un état de signal identique.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	1	-	-	-	-	0	1	1	1

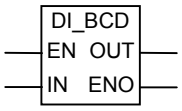
Exemple



La conversion est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le contenu du mot de mémento MW10 est lu comme nombre entier de 16 bits et converti en nombre entier de 32 bits. Le résultat est rangé dans MD12. La sortie A 4.0 est mise à 1 si la conversion est exécutée (ENO = EN).

3.6 DI_BCD : Convertir entier de 32 bits en nombre DCB

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN	DINT	E, A, M, D, L ou constante	Nombre entier de 32 bits
OUT	DWORD	E, A, M, D, L	Valeur DCB du nombre entier de 32 bits
ENO	BOOL	E, A, M, D, L	Sortie de validation

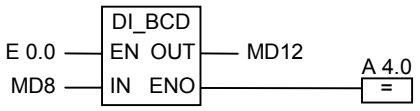
Description

L'opération **Convertir entier de 32 bits en nombre DCB** lit le contenu du paramètre d'entrée IN comme nombre entier de 32 bits, le convertit en un nombre décimal codé binaire à sept chiffres (DCB, ± 9 999 999) et range le résultat dans le paramètre de sortie OUT. En cas de débordement, ENO est mis à 0.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	X	-	-	X	X	0	X	X	1

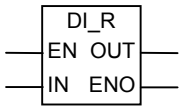
Exemple



La conversion est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le contenu du double mot de mémento MD8 est lu comme nombre entier de 32 bits et converti en nombre DCB à sept chiffres. Le résultat est rangé dans MD12. En cas de débordement, la sortie A 4.0 est mise à 0. Si l'état de signal à l'entrée EN est égal à 0 (la conversion n'est pas exécutée), l'état de signal à la sortie A 4.0 est également 0.

3.7 DI_R : Convertir entier de 32 bits en nombre réel

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN	DINT	E, A, M, D, L ou constante	Valeur à convertir
OUT	REAL	E, A, M, D, L	Résultat
ENO	BOOL	E, A, M, D, L	Sortie de validation

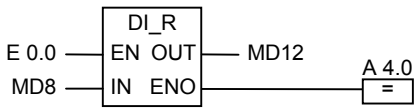
Description

L'opération **Convertir entier de 32 bits en nombre réel** lit le contenu du paramètre d'entrée IN comme nombre entier de 32 bits, le convertit en un nombre réel et range le résultat dans le paramètre de sortie OUT. ENO et EN ont toujours un état de signal identique.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	1	-	-	-	-	0	1	1	1

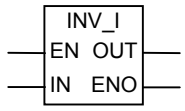
Exemple



La conversion est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le contenu du double mot de mémento MD8 est lu comme nombre entier de 32 bits et converti en nombre réel. Le résultat est rangé dans MD12. La sortie A 4.0 est mise à 0 si la conversion n'est pas exécutée (ENO = EN).

3.8 INV_I : Complément à 1 d'entier de 16 bits

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN	INT	E, A, M, D, L ou constante	Valeur d'entrée
OUT	INT	E, A, M, D, L	Complément à 1 du nombre entier de 16 bits
ENO	BOOL	E, A, M, D, L	Sortie de validation

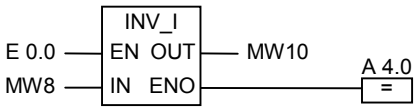
Description

L'opération **Complément à 1 d'entier de 16 bits** lit le contenu du paramètre d'entrée IN et exécute l'opération de combinaison **OU exclusif mot** avec le masque hexadécimal FFFF afin d'inverser la valeur de chaque bit. Le résultat est rangé dans le paramètre de sortie OUT. ENO et EN ont toujours un état de signal identique.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	1	-	-	-	-	0	1	1	1

Exemple



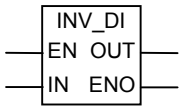
La conversion est exécutée si l'état de signal est 1 à l'entrée E 0.0. Chaque bit de MW8 est inversé :

MW8 = 01000001 10000001 -> MW10 = 10111110 01111110

Si E 0.0 est égal à 0, la conversion n'est pas exécutée et la sortie A 4.0 est mise à 0 (ENO = EN).

3.9 INV_DI : Complément à 1 d'entier de 32 bits

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN	DINT	E, A, M, D, L ou constante	Valeur d'entrée
OUT	DINT	E, A, M, D, L	Complément à 1 du nombre entier de 32 bits
ENO	BOOL	E, A, M, D, L	Sortie de validation

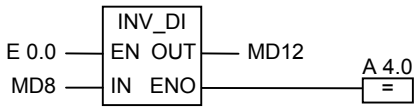
Description

L'opération **Complément à 1 d'entier de 32 bits** lit le contenu du paramètre d'entrée IN et exécute l'opération de combinaison **OU exclusif double mot** avec le masque hexadécimal FFFF FFFF afin d'inverser la valeur de chaque bit. Le résultat est rangé dans le paramètre de sortie OUT. ENO et EN ont toujours un état de signal identique.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	1	-	-	-	-	0	1	1	1

Exemple



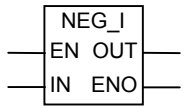
La conversion est exécutée si l'état de signal est 1 à l'entrée E 0.0. Chaque bit du double mot de mémento MD8 est inversé :

MD8 = F0FF FFF0 -> MD12 = 0F00 000F

Si E 0.0 est égal à 0, la conversion n'est pas exécutée et la sortie A 4.0 est mise à 0 (ENO = EN).

3.10 NEG_I : Complément à 2 d'entier de 16 bits

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN	INT	E, A, M, D, L ou constante	Valeur d'entrée
OUT	INT	E, A, M, D, L	Complément à 2 du nombre entier de 16 bits
ENO	BOOL	E, A, M, D, L	Sortie de validation

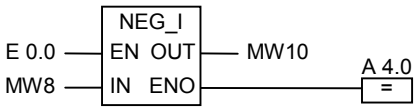
Description

L'opération **Complément à 2 d'entier de 16 bits** lit le contenu du paramètre d'entrée IN et en change le signe (par exemple, valeur positive en valeur négative). Le résultat est rangé dans le paramètre de sortie OUT. ENO et EN ont toujours un état de signal identique. Exception : si l'état de signal de EN est égal à 1 et qu'il y ait débordement, l'état de signal de ENO est égal à 0.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	X	X	X	X	X	0	X	X	1

Exemple



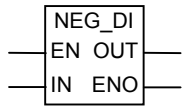
La conversion est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le contenu du mot de mémento MW8 est rangé dans le paramètre de sortie OUT - mot de mémento MW10 - avec le signe opposé.

MW8 = +10 -> MW10 = - 10

Si l'état de signal de EN est égal à 1 et qu'il y ait débordement, l'état de signal de ENO ainsi que celui de la sortie A 4.0 sont égaux à 0. La sortie A 4.0 est mise à 0 si la conversion n'est pas exécutée (ENO = EN).

3.11 NEG_DI : Complément à 2 d'entier de 32 bits

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN	DINT	E, A, M, D, L ou constante	Valeur d'entrée
OUT	DINT	E, A, M, D, L	Complément à 2 du nombre entier de 32 bits
ENO	BOOL	E, A, M, D, L	Sortie de validation

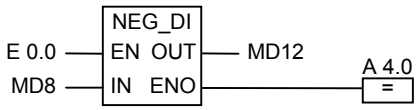
Description

L'opération **Complément à 2 d'entier de 32 bits** lit le contenu du paramètre d'entrée IN et en change le signe (par exemple, valeur positive en valeur négative). Le résultat est rangé dans le paramètre de sortie OUT. ENO et EN ont toujours un état de signal identique. Exception : si l'état de signal de EN est égal à 1 et qu'il y ait débordement, l'état de signal de ENO est égal à 0.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	X	X	X	X	X	0	X	X	1

Exemple



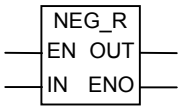
La conversion est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le contenu du double mot de mémento MD8 est rangé dans le paramètre de sortie OUT - double mot de mémento MD12 - avec le signe opposé.

MD8 = + 60.000 -> MD12 = - 60.000.

Si l'état de signal de EN est égal à 1 et qu'il y ait débordement, l'état de signal de ENO ainsi que celui de la sortie A 4.0 sont égaux à 0. La sortie A 4.0 est mise à 0 si la conversion n'est pas exécutée (ENO = EN).

3.12 NEG_R : Inverser le signe d'un nombre réel

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN	REAL	E, A, M, D, L ou constante	Valeur d'entrée
OUT	REAL	E, A, M, D, L	Le résultat est la négation de la valeur d'entrée.
ENO	BOOL	E, A, M, D, L	Sortie de validation

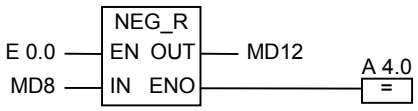
Description

L'opération **Inverser le signe d'un nombre réel** lit le contenu du paramètre d'entrée IN et en change le bit de signe (par exemple, de 0 pour une valeur positive en 1 pour une valeur négative). Les bits de l'exposant et de la mantisse restent inchangés. Le résultat est rangé dans le paramètre de sortie OUT. ENO et EN ont toujours un état de signal identique. Exception : si l'état de signal de EN est égal à 1 et qu'il y ait débordement, l'état de signal de ENO est égal à 0.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	X	-	-	-	-	0	X	X	1

Exemple



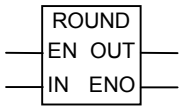
La conversion est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le contenu du double mot de mémento MD8 est rangé dans le paramètre de sortie OUT - double mot de mémento MD12 - avec le signe opposé.

MD8 = + 6,234 -> MD12 = - 6,234

La sortie A 4.0 est mise à 0 si la conversion n'est pas exécutée (ENO = EN).

3.13 ROUND : Arrondir à entier de 32 bits

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN	REAL	E, A, M, D, L ou constante	Valeur à arrondir
OUT	DINT	E, A, M, D, L	IN arrondi au nombre entier le plus proche
ENO	BOOL	E, A, M, D, L	Sortie de validation

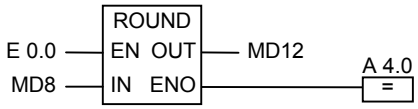
Description

L'opération **Arrondir à entier de 32 bits** lit le contenu du paramètre d'entrée IN comme nombre réel et le convertit en nombre entier de 32 bits. Le résultat, qui est le nombre entier le plus proche, est rangé dans le paramètre de sortie OUT. Si la partie fractionnaire est égale à "x,5", le résultat fourni est le nombre pair (exemple : 2,5 -> 2 et 1,5 -> 2). En cas de débordement, ENO est mis à 0. Si l'entrée n'est pas un nombre réel, les bits de débordement et de débordement mémorisé (DEB et DM) prennent la valeur 1 et ENO est égal à 0.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	X	-	-	X	X	0	X	X	1

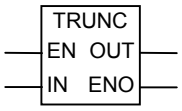
Exemple



La conversion est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le contenu du double mot de mémento MD8 est lu comme nombre réel et converti en nombre entier de 32 bits selon le principe "arrondi au plus proche". Le résultat est rangé dans MD12. En cas de débordement, la sortie A 4.0 est mise à 0. Si l'état de signal de l'entrée EN est égal à 0 (conversion non exécutée), l'état de signal de la sortie A 4.0 est également 0.

3.14 TRUNC : Tronquer à la partie entière (32 bits)

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN	REAL	E, A, M, D, L ou constante	Valeur à tronquer
OUT	DINT	E, A, M, D, L	Partie entière de IN
ENO	BOOL	E, A, M, D, L	Sortie de validation

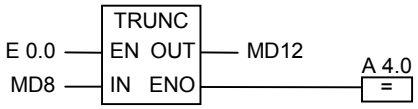
Description

L'opération **Tronquer à la partie entière (32 bits)** lit le contenu du paramètre d'entrée IN comme nombre réel et le convertit en nombre entier de 32 bits (exemple : 1,5 devient 1). Le résultat, qui est la partie entière du nombre réel, est rangé dans le paramètre de sortie OUT. En cas de débordement, ENO est mis à 0. Si l'entrée n'est pas un nombre réel, les bits de débordement et de débordement mémorisé (DEB et DM) prennent la valeur 1 et ENO est égal à 0.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	X	-	-	X	X	0	X	X	1

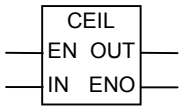
Exemple



La conversion est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le contenu du double mot de mémento MD8 est lu comme nombre réel et converti en nombre entier de 32 bits selon le principe "arrondi à zéro". Le résultat, qui est la partie entière du nombre réel, est rangé dans MD12. En cas de débordement, la sortie A 4.0 est mise à 0. Si l'état de signal de l'entrée EN est égal à 0 (conversion non exécutée), l'état de signal de la sortie A 4.0 est également 0.

3.15 CEIL : Convertir nombre réel en entier supérieur le plus proche

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN	REAL	E, A, M, D, L ou constante	Valeur à convertir
OUT	DINT	E, A, M, D, L	Résultat
ENO	BOOL	E, A, M, D, L	Sortie de validation

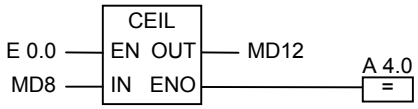
Description

L'opération **Convertir nombre réel en entier supérieur le plus proche** lit le contenu du paramètre d'entrée IN comme nombre réel et le convertit en un nombre entier de 32 bits (exemple : +1,2 -> +2 et -1,5 -> -1). Le résultat, qui est le nombre entier supérieur le plus proche du nombre réel indiqué, est rangé dans le paramètre de sortie OUT. En cas de débordement, ENO est mis à 0. Si l'entrée n'est pas un nombre réel, les bits de débordement et de débordement mémorisé (DEB et DM) prennent la valeur 1 et ENO est égal à 0.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	X	-	-	X	X	0	X	X	1

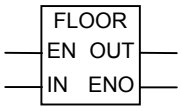
Exemple



La conversion est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le contenu du double mot de mémento MD8 est lu comme nombre réel et converti en nombre entier de 32 bits selon le principe "arrondi à plus l'infini". Le résultat est rangé dans MD12. En cas de débordement, la sortie A 4.0 est mise à 0. Si l'état de signal de l'entrée EN est égal à 0 (conversion non exécutée), l'état de signal de la sortie A 4.0 est également 0.

3.16 FLOOR : Convertir nombre réel en entier inférieur le plus proche

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN	REAL	E, A, M, D, L ou constante	Valeur à convertir
OUT	DINT	E, A, M, D, L	Résultat
ENO	BOOL	E, A, M, D, L	Sortie de validation

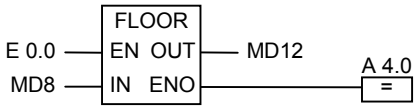
Description

L'opération **Convertir nombre réel en entier inférieur le plus proche** lit le contenu du paramètre d'entrée IN comme nombre réel et le convertit en un nombre entier de 32 bits (exemple : +1,5 -> +1 et -1,5 -> -2). Le résultat, qui est le nombre entier inférieur le plus proche du nombre réel indiqué, est rangé dans le paramètre de sortie OUT. En cas de débordement, ENO est mis à 0. Si l'entrée n'est pas un nombre réel, les bits de débordement et de débordement mémorisé (DEB et DM) prennent la valeur 1 et ENO est égal à 0.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	X	-	-	X	X	0	X	X	1

Exemple



La conversion est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le contenu du double mot de mémento MD8 est lu comme nombre réel et converti en nombre entier de 32 bits selon le principe "arrondi à moins l'infini". Le résultat est rangé dans MD12. En cas de débordement, la sortie A 4.0 est mise à 0. Si l'état de signal de l'entrée EN est égal à 0 (conversion non exécutée), l'état de signal de la sortie A 4.0 est également 0.

4 Opérations de comptage

4.1 Vue d'ensemble des opérations de comptage

Zone de mémoire

Une zone de mémoire est réservée aux compteurs dans votre CPU. Un mot de 16 bits y est réservé pour chaque compteur. La programmation en LOG permet d'utiliser jusqu'à 256 compteurs.

Les opérations de comptage sont les seules fonctions à avoir accès à la zone de mémoire réservée aux compteurs.

Valeur de comptage

La valeur de comptage est contenue dans les bits 0 à 9 du mot de comptage. Lorsque le compteur est mis à 1, la valeur que vous avez définie y est placée par l'accumulateur. La plage de la valeur de comptage est comprise entre 0 et 999.

Vous pouvez modifier cette valeur en utilisant les opérations :

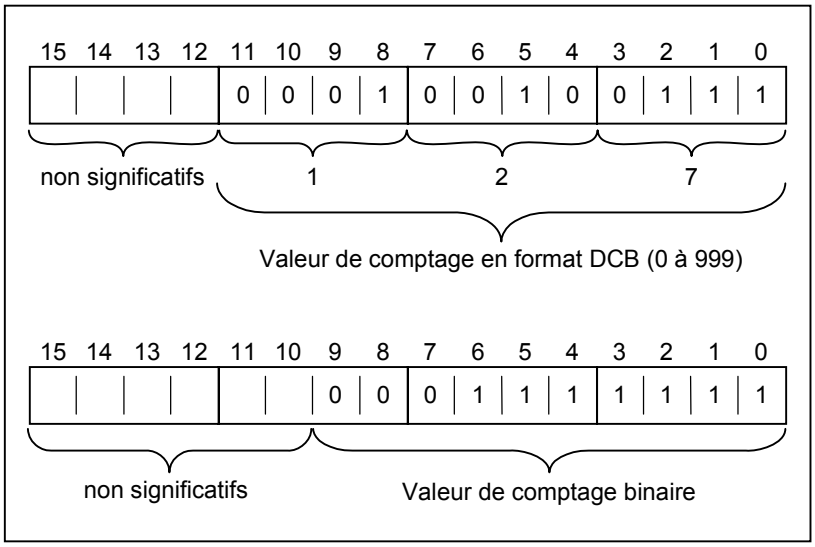
- ZAEHLER : Paramétrage et compteur incrémental/décrémental
- Z_VORW : Paramétrage et compteur incrémental
- Z_RUECK : Paramétrage et compteur décrémental
- SZ : Initialiser compteur
- ZV : Incrémenter
- ZR : Décrémenter

Configuration des bits dans le compteur

Pour assigner une valeur initiale à un compteur, vous chargez un nombre compris entre 0 et 999, par exemple 127, au format suivant comme valeur de comptage : C# 127. C# correspond au format décimal codé binaire.

Les bits 0 à 11 du compteur contiennent la valeur de comptage en format DCB, c´est -à-dire chaque groupe de quatre bits contient le code binaire d'une valeur décimale.

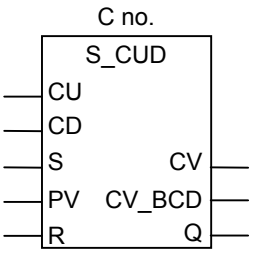
La figure suivante montre le contenu du compteur après le chargement de la valeur 127, ainsi que le contenu de la cellule de compteur après assignation d'une valeur.



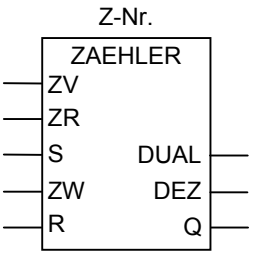
4.2 ZAEHLER : Paramétrage et compteur incrémental/décrémental

Représentation

Anglaise



Allemande



Paramètre Anglaise	Paramètre Allemande	Type de données	Zone de mémoire	Description
C no.	Z Nr.	COUNTER	Z	Numéro d'identification du compteur. La plage dépend de la CPU.
CU	ZV	BOOL	E, A, M, D, L, T, Z	Entrée d'incrémentation
CD	ZR	BOOL	E, A, M, D, L	Entrée de décrémentation
S	S	BOOL	E, A, M, D, L	Entrée d'initialisation du compteur
PV	ZW	WORD	E, A, M, D, L ou constante	Valeur de comptage comprise entre 0 et 999 ou valeur de comptage saisie en format DCB sous la forme C#<valeur>
R	R	BOOL	E, A, M, D, L,T, Z	Entrée de remise à zéro
CV	DUAL	WORD	E, A, M, D, L	Valeur de comptage en cours (format hexadécimal)
CV_BCD	DEZ	WORD	E, A, M, D, L	Valeur de comptage en cours (format DCB)
Q	Q	BOOL	E, A, M, D, L	Etat du compteur

Description

Un front montant (transition de 0 à 1) à l'entrée S de l'opération **Paramétrage et compteur incrémental/décrémental** initialise le compteur à la valeur figurant dans l'entrée ZW. La valeur du compteur est incrémentée d'une unité en cas de front montant à l'entrée ZV si la valeur de comptage est inférieure à 999 ; elle est décrémentée d'une unité en cas de front montant à l'entrée ZR si la valeur de comptage est supérieure à 0. En cas de front montant aux deux entrées de comptage, les deux opérations sont traitées et la valeur de comptage reste inchangée.

Si le compteur est mis à 1 et si le RLG = 1 aux entrées ZV/ZR, le compteur compte une fois dans le cycle **suivant**, même si aucun changement de front n'a eu lieu.

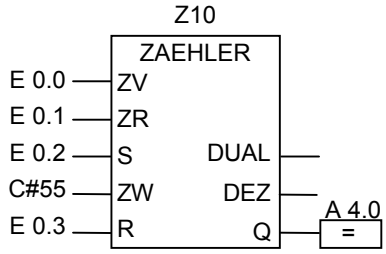
La valeur 1 à l'entrée R remet le compteur à zéro ; la valeur de comptage prend la valeur nulle.

L'interrogation à 1 de l'état de signal à la sortie Q donne 1 comme résultat lorsque la valeur de comptage est supérieure à 0 ; cette interrogation donne 0 comme résultat lorsque la valeur de comptage est égale à 0.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	-	-	-	-	X	X	X	1

Exemple



Le compteur Z10 est initialisé à la valeur 55 si l'état de signal passe de 0 à 1 à l'entrée E 0.2. Si l'état de signal de E 0.0 passe de 0 à 1, la valeur du compteur Z10 est incrémentée d'un à moins qu'elle ne soit déjà égale à 999. Si l'état de signal de E 0.1 passe de 0 à 1, la valeur du compteur Z10 est décrémentée d'un à moins qu'elle ne soit déjà égale à 0. Si l'état de signal de l'entrée E 0.3 passe de 0 à 1, la valeur de comptage de Z10 est mise à zéro. La sortie A 4.0 est à 1 tant que Z10 est différent de zéro.

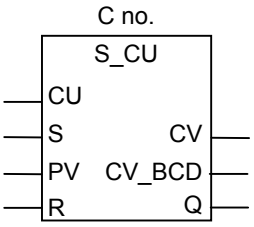
Nota

Il est recommandé d'utiliser un compteur à un seul emplacement dans le programme (risque d'erreurs de comptage).

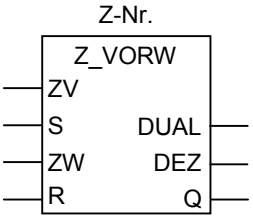
4.3 Z_VORW : Paramétrage et compteur incrémental

Représentation

Anglaise



Allemande



Paramètre Anglaise	Paramètre Allemande	Type de données	Zone de mémoire	Description
C no.	Z Nr.	COUNTER	Z	Numéro d'identification du compteur. La plage dépend de la CPU.
CU	ZV	BOOL	E, A, M, D, L	Entrée d'incrémentation
S	S	BOOL	E, A, M, D, L, T, Z	Entrée d'initialisation du compteur
PV	ZW	WORD	E, A, M, D, L ou constante	Valeur de comptage comprise entre 0 et 999 ou valeur de comptage saisie en format DCB sous la forme C#<valeur>
R	R	BOOL	E, A, M, D, L, T, Z	Entrée de remise à zéro
CV	DUAL	WORD	E, A, M, D, L	Valeur de comptage en cours (format hexadécimal)
CV_BCD	DEZ	WORD	E, A, M, D, L	Valeur de comptage en cours (format DCB)
Q	Q	BOOL	E, A, M, D, L	Etat du compteur

Description

Un front montant (transition de 0 à 1) à l'entrée S de l'opération **Paramétrage et compteur incrémental** initialise le compteur à la valeur figurant dans l'entrée ZW. La valeur du compteur est incrémentée d'une unité en cas de front montant à l'entrée ZV si la valeur de comptage est inférieure à 999.

Si le compteur est mis à 1 et si le RLG = 1 à l'entrée ZV, le compteur compte une fois dans le cycle **suivant**, même si aucun changement de front n'a eu lieu.

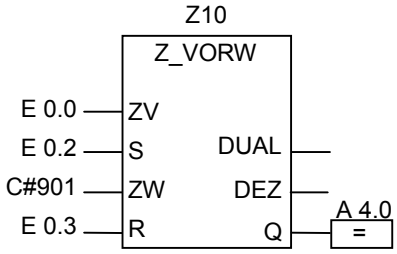
La valeur 1 à l'entrée R remet le compteur à zéro ; la valeur de comptage prend la valeur nulle.

L'interrogation à 1 de l'état de signal à la sortie Q donne 1 comme résultat lorsque la valeur de comptage est supérieure à 0 ; cette interrogation donne 0 comme résultat lorsque la valeur de comptage est égale à 0.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	-	-	-	-	X	X	X	1

Exemple



Le compteur Z10 est initialisé à la valeur 901 si l'état de signal passe de 0 à 1 à l'entrée E 0.2. Si l'état de signal de E 0.0 passe de 0 à 1, la valeur du compteur Z10 est incrémentée d'un à moins qu'elle ne soit déjà égale à 999. Si l'état de signal de l'entrée E 0.3 passe de 0 à 1, la valeur de comptage de Z10 est mise à zéro. La sortie A 4.0 est à 1 tant que Z10 est différent de zéro.

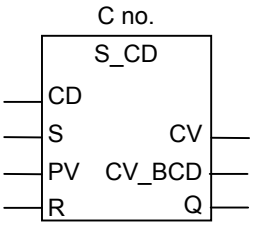
Nota

Il est recommandé d'utiliser un compteur à un seul emplacement dans le programme (risque d’erreurs de comptage).

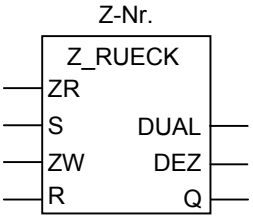
4.4 Z_RUECK : Paramétrage et compteur décrémental

Représentation

Anglaise



Allemande



Paramètre Anglaise	Paramètre Allemande	Type de données	Zone de mémoire	Description
C no.	Z Nr.	COUNTER	Z	Numéro d'identification du compteur. La plage dépend de la CPU.
CD	ZR	BOOL	E, A, M, D, L	Entrée de décrémentation
S	S	BOOL	E, A, M, D, L, T, Z	Entrée d'initialisation du compteur
PV	ZW	WORD	E, A, M, D, L ou constante	Valeur de comptage comprise entre 0 et 999 ou valeur de comptage saisie en format DCB sous la forme C#<valeur>
R	R	BOOL	E, A, M, D, L, T, Z	Entrée de remise à zéro
CV	DUAL	WORD	E, A, M, D, L	Valeur de comptage en cours (format hexadécimal)
CV_BCD	DEZ	WORD	E, A, M, D, L	Valeur de comptage en cours (format DCB)
Q	Q	BOOL	E, A, M, D, L	Etat du compteur

Description

Un front montant (transition de 0 à 1) à l'entrée S de l'opération **Paramétrage et compteur décrémental** initialise le compteur à la valeur figurant dans l'entrée ZW. La valeur du compteur est décrémentée d'une unité en cas de front montant à l'entrée ZR si la valeur de comptage est supérieure à 0.

Si le compteur est mis à 1 et si le RLG = 1 à l'entrée ZR, le compteur compte une fois dans le cycle **suivant**, même si aucun changement de front n'a eu lieu.

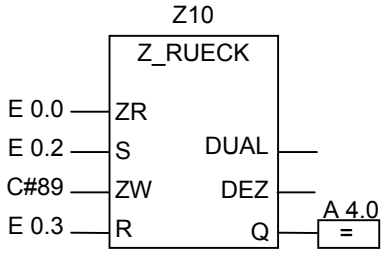
La valeur 1 à l'entrée R remet le compteur à zéro ; la valeur de comptage prend la valeur nulle.

L'interrogation à 1 de l'état de signal à la sortie Q donne 1 comme résultat lorsque la valeur de comptage est supérieure à 0 ; cette interrogation donne 0 comme résultat lorsque la valeur de comptage est égale à 0.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	-	-	-	-	X	X	X	1

Exemple



Le compteur Z10 est initialisé à la valeur 89 si l'état de signal passe de 0 à 1 à l'entrée E 0.2. Si l'état de signal de E 0.0 passe de 0 à 1, la valeur du compteur Z10 est décrémentée d'un à moins qu'elle ne soit déjà égale à 0. Si l'état de signal de l'entrée E 0.3 passe de 0 à 1, la valeur de comptage de Z10 est mise à zéro.

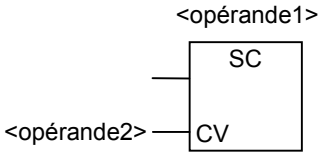
Nota

Il est recommandé d'utiliser un compteur à un seul emplacement dans le programme (risque d'erreurs de comptage).

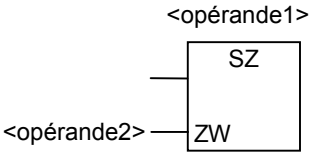
4.5 SZ : Initialiser compteur

Représentation

Anglaise



Allemande



Paramètre Anglaise	Paramètre Allemande	Type de données	Zone de mémoire	Description
Numéro de compteur	Numéro de compteur	COUNTER	Z	<opérande1> indique le numéro du compteur qui doit être initialisé.
CV	ZW	WORD	E, A, M, D, L ou constante	La valeur d'initialisation (opérande2) peut être comprise entre 0 et 999. Pour une constante, elle doit être précédée de C# afin d'indiquer le format DCB, par exemple C#100.

Description

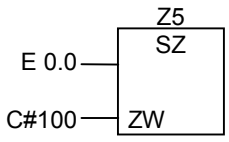
L'opération **Initialiser compteur** vous permet d'affecter une valeur initiale au compteur que vous avez défini. Elle ne s'exécute que si le RLG présente un front montant (c'est-à-dire s'il passe de 0 à 1).

Vous ne pouvez placer la boîte **Initialiser compteur** qu'à l'extrémité droite de la séquence combinatoire. Il est toutefois possible d'en utiliser plusieurs.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	-	-	-	-	0	-	-	0

Exemple



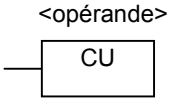
Si l'état de signal à l'entrée E 0.0 passe de 0 à 1 (front montant du RLG), le compteur Z5 est initialisé avec la valeur 100. C# indique que vous entrez une valeur en format DCB.

En l'absence de front montant, la valeur de Z5 reste inchangée.

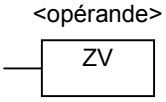
4.6 ZV : Incrémenter

Représentation

Anglaise



Allemande



Paramètre	Type de données	Zone de mémoire	Description
Numéro de compteur	COUNTER	Z	L'opérande indique le numéro du compteur à incrémenter.

Description

L'opération **Incrémenter** incrémente d'un la valeur du compteur précisé si le RLG présente un front montant (c'est-à-dire s'il passe de 0 à 1) et si la valeur du compteur est inférieure à 999. En l'absence de front montant au RLG ou si le compteur est déjà égal à 999, la valeur du compteur reste inchangée.

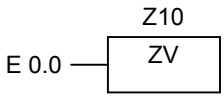
L'opération **Initialiser compteur** permet d'initialiser le compteur.

Vous ne pouvez placer la boîte **Incrémenter** qu'à l'extrémité droite de la séquence combinatoire. Il est toutefois possible d'en utiliser plusieurs.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	-	-	-	-	0	-	-	0

Exemple



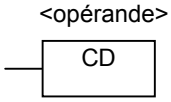
Si l'état de signal en E 0.0 passe de 0 à 1 (front montant du RLG), la valeur du compteur Z10 est incrémentée de 1, à moins qu'elle ne soit déjà égale à 999.

En l'absence de front montant, la valeur de Z10 reste inchangée.

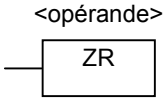
4.7 ZR : Décrémenter

Représentation

Anglaise



Allemande



Paramètre	Type de données	Zone de mémoire	Description
Numéro de compteur	COUNTER	Z	L'opérande indique le numéro du compteur à décrémenter.

Description

L'opération **Décrémenter** décrémente d'un la valeur du compteur précisé si le RLG présente un front montant (c'est-à-dire s'il passe de 0 à 1) et si la valeur du compteur est supérieure à 0. En l'absence de front montant au RLG ou si le compteur est déjà égal à 0, la valeur du compteur reste inchangée.

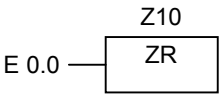
L'opération **Initialiser compteur** permet d'initialiser le compteur (voir aussi la rubrique Initialiser compteur).

Vous ne pouvez placer la boîte **Décrémenter** qu'à l'extrémité droite de la séquence combinatoire. Il est toutefois possible d'en utiliser plusieurs.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	-	-	-	-	0	-	-	0

Exemple



Si l'état de signal en E 0.0 passe de 0 à 1 (front montant du RLG), la valeur du compteur Z10 est décrémentée de 1, à moins qu'elle ne soit déjà égale à 0.

En l'absence de front montant, la valeur de Z10 reste inchangée.

5 Opérations sur blocs de données

5.1 OPN : Ouvrir bloc de données

Représentation

<DB numéro> ou
<DI numéro>

OPN

Paramètre	Type de données	Zone de mémoire	Description
Numéro du DB ou du DI	-	-	Numéro du DB ou du DI. La plage autorisée dépend de votre CPU.

Description

L'opération **Ouvrir bloc de données** ouvre un bloc de données (DB) ou un bloc de données d'instance (DI). Le numéro du bloc de données est transféré dans le registre DB ou DI. Les commandes DB et DI suivantes accèdent aux blocs de données correspondant au contenu de ces registres.

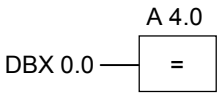
Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	-	-	-	-	-	-	-	-

Exemple

Réseau 1

Réseau 2



Le DB10 est le DB ouvert en cours. C'est pourquoi l'interrogation en DBX 0.0 se réfère au bit 0 de l'octet de données 0 du bloc de données DB10. L'état de signal de ce bit est affecté à la sortie A 4.0.

6 Opérations de saut

6.1 Vue d'ensemble des opérations de saut

Description

Vous pouvez utiliser cette opération dans tous les blocs de code, à savoir les blocs d'organisation (OB), les blocs fonctionnels (FB) et les fonctions (FC).

Vous disposez des opérations de saut suivantes :

- JMP : Saut inconditionnel
- JMPN : Saut si 0 (conditionnel)
- JMP : Saut si 1 (conditionnel)

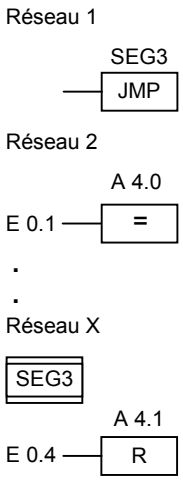
Repère de saut comme opérande

L'opérande d'une opération de saut est un repère de saut qui comporte 4 caractères au maximum. Le premier caractère doit être une lettre, les autres caractères pouvant être des lettres ou des chiffres (par exemple, SEG3). Le repère de saut indique l'endroit où doit sauter le programme.

Vous précisez le repère de saut au-dessus de la boîte JMP.

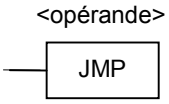
Repère de saut comme destination

Le repère de saut de destination doit se trouver au début d'un réseau. Pour l'indiquer, sélectionnez REPERE dans la boîte de sélection LOG et saisissez son nom dans la boîte vide qui apparaît alors.



6.2 JMP : Saut inconditionnel

Représentation



Paramètre	Type de données	Zone de mémoire	Description
Nom d'un repère de saut	-	-	L'opérande indique le repère vers lequel doit se faire le saut inconditionnel.

Description

L'opération **Saut inconditionnel** correspond à une opération **Aller à un repère de saut**. Aucune des opérations entre l'opération de saut et le repère de saut n'est exécutée.

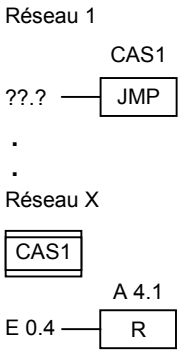
Vous pouvez utiliser cette opération dans tous les blocs de code, à savoir les blocs d'organisation (OB), les blocs fonctionnels (FB) et les fonctions (FC).

Il ne doit pas y avoir de combinaison en amont de la boîte **Saut inconditionnel**.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	-	-	-	-	-	-	-	-

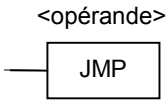
Exemple



Le saut est exécuté à chaque fois. Aucune des opérations entre l'opération de saut et le repère de saut n'est exécutée.

6.3 JMP : Saut si 1 (conditionnel)

Représentation



Paramètre	Type de données	Zone de mémoire	Description
Nom d'un repère de saut	-	-	L'opérande indique le repère vers lequel doit se faire le saut si le RLG est égal à 1.

Description

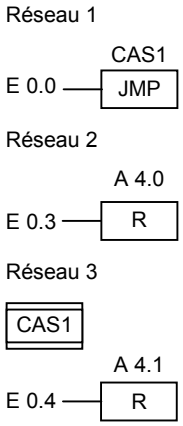
L'opération **Saut si 1 (conditionnel)** correspond à une opération **Aller à un repère de saut** si le RLG est égal à 1. Pour cette opération, utilisez l'élément LOG Saut inconditionnel, mais avec une combinaison en amont. Le saut conditionnel n'est exécuté que si le résultat logique de cette combinaison est égal à 1. Dans ce cas, aucune des opérations entre l'opération de saut et le repère de saut n'est exécutée.

Vous pouvez utiliser cette opération dans tous les blocs de code, à savoir les blocs d'organisation (OB), les blocs fonctionnels (FB) et les fonctions (FC).

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	-	-	-	-	0	1	1	0

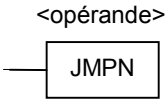
Exemple



Le saut au repère CAS1 est exécuté si l'état de signal est 1 à l'entrée E 0.0. L'opération de mise à zéro de la sortie A 4.0 n'est pas exécutée même si l'état de signal est 1 à l'entrée E 0.3.

6.4 JMPN : Saut si 0 (conditionnel)

Représentation



Paramètre	Type de données	Zone de mémoire	Description
Nom d'un repère de saut	-	-	L'opérande indique le repère vers lequel doit se faire le saut si le RLG est égal à 0.

Description

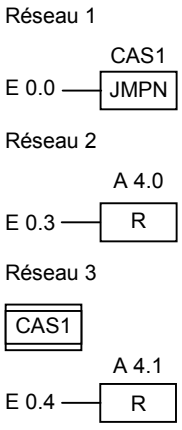
L'opération **Saut si 0 (conditionnel***)* correspond à une opération **Aller à un repère de saut** si le RLG est égal à 0.

Vous pouvez utiliser cette opération dans tous les blocs de code, à savoir les blocs d'organisation (OB), les blocs fonctionnels (FB) et les fonctions (FC).

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	-	-	-	-	0	1	1	0

Exemple



Le saut au repère CAS1 est exécuté si l'état de signal est 0 à l'entrée E 0.0. L'opération de mise à zéro de la sortie A 4.0 n'est pas exécutée même si l'état de signal est 1 à l'entrée E 0.3.
Aucune des opérations entre l'opération de saut et le repère de saut n'est exécutée.

6.5 LABEL : Repère de saut

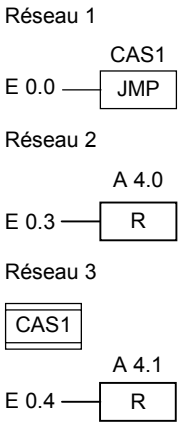
Représentation

Description

Le repère de saut identifie la destination d'une opération de saut. Le repère de saut consiste de 4 caractères alphanumériques, le premier devant être une lettre, par exemple CAS1.

Il doit exister un repère de saut (LABEL) pour chaque saut conditionnel (**JMP** ou **JMPN**).

Exemple



Le saut au repère CAS1 est exécuté si l'état de signal est 1 à l'entrée E 0.0.
En raison du saut, l'opération de mise à zéro de la sortie A 4.0 n'est pas exécutée même si l'état de signal est 1 à l'entrée E 0.3.

7 Opérations arithmétiques sur nombres entiers

7.1 Vue d'ensemble des opérations arithmétiques sur nombre entiers

Description

Les opérations arithmétiques sur nombres entiers permettent d'exécuter les fonctions arithmétiques suivantes sur **deux** nombres entiers (16 et 32 bits) :

- ADD_I : Additionner entiers de 16 bits
- SUB_I : Soustraire entiers de 16 bits
- MUL_I : Multiplier entiers de 16 bits
- DIV_I : Diviser entiers de 16 bits
- ADD_DI : Additionner entiers de 32 bits
- SUB_DI : Soustraire entiers de 32 bits
- MUL_DI : Multiplier entiers de 32 bits
- DIV_DI : Diviser entiers de 32 bits
- MOD_DI : Reste de division (32 bits)

Voir aussi Evaluation des bits du mot d'état dans les opérations sur nombres entiers

7.2 Evaluation des bits du mot d'état pour les opérations sur nombres entiers

Description

Les opérations arithmétiques sur nombres entiers affectent les bits suivants du mot d'état :

- BI1 et BI0,
- DEB
- et DM.

Les tableaux ci-dessous montrent l'état de signal des bits du mot d'état pour les résultats d'opérations sur nombres entiers (16 et 32 bits) :

Plage autorisée	**BI1**	**BI0**	**DEB**	**DM**
0 (zéro)	0	0	0	*
16 bits : -32 768 <= résultat < 0 (nombre négatif) 32 bits : -2 147 483 648 <= résultat < 0 (nombre négatif)	0	1	0	*
16 bits : 32 767 > =résultat > 0 (nombre positif) 32 bits : 2 147 483 647 > = résultat > 0 (nombre positif)	1	0	0	*

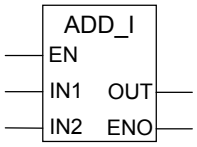
* Le bit DM n'est pas influencé par le résultat de l'opération.

Plage non autorisée	**BI1**	**BI0**	**DEB**	**DM**
Dépassement négatif de la plage pour une addition 16 bits : résultat = -65536 32 bits : résultat = -4 294 967 296	0	0	1	1
Dépassement négatif de la plage pour une multiplication 16 bits : résultat < -32 768 (nombre négatif) 32 bits : résultat < -2 147 483 648 (nombre négatif)	0	1	1	1
Dépassement positif de la plage pour addition, soustraction 16 bits : résultat > 32 767 (nombre positif) 32 bits : résultat > 2 147 483 647 (nombre positif)	0	1	1	1
Dépassement positif de la plage pour multiplication, division 16 bits : résultat > 32 767 (nombre positif) 32 bits : résultat > 2 147 483 647 (nombre positif)	1	0	1	1
Dépassement négatif de la plage pour addition, soustraction 16 bits : résultat < -32 768 (nombre négatif) 32 bits : résultat < -2 147 483 648 (nombre négatif)	1	0	1	1
Division par zéro	1	1	1	1

Opération	**BI1**	**BI0**	**DEB**	**DM**
+D : résultat = -4 294 967 296	0	0	1	1
/D ou MOD : division par 0	1	1	1	1

7.3 ADD_I : Additionner entiers de 16 bits

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN1	INT	E, A, M, D, L ou constante	Premier terme de l'addition
IN2	INT	E, A, M, D, L ou constante	Second terme de l'addition
OUT	INT	E, A, M, D, L	Résultat de l'addition
ENO	BOOL	E, A, M, D, L	Sortie de validation

Description

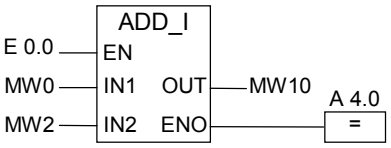
L'opération **Additionner entiers de 16 bits** est activée si l'état de signal est 1 à l'entrée de validation EN. Cette opération additionne les entrées IN1 et IN2 et range le résultat dans la sortie OUT. Si ce résultat est hors de la plage autorisée pour un nombre entier de 16 bits, les bits de débordement et de débordement mémorisé (DEB et DM) sont mis à 1 et la sortie ENO est mise à 0.

Voir aussi Evaluation des bits du mot d'état pour les opérations sur nombres entiers.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	X	X	X	X	X	0	X	X	1

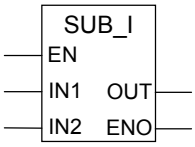
Exemple



La boîte ADD_I est activée si l'état de signal est 1 à l'entrée E 0.0. Le résultat de l'addition MW0 + MW2 est rangé dans le mot de mémento MW10. Si ce résultat est hors de la plage autorisée pour un nombre entier de 16 bits ou si l'état de signal de l'entrée E 0.0 est égal à 0, la sortie A 4.0 est mise à 0 et l'opération n'est pas exécutée.

7.4 SUB_I : Soustraire entiers de 16 bits

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN1	INT	E, A, M, D, L ou constante	Valeur dont on soustrait
IN2	INT	E, A, M, D, L ou constante	Valeur à soustraire
OUT	INT	E, A, M, D, L	Résultat de la soustraction
ENO	BOOL	E, A, M, D, L	Sortie de validation

Description

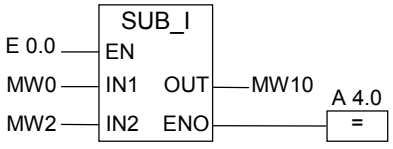
L'opération **Soustraire entiers de 16 bits** est activée si l'état de signal est 1 à l'entrée de validation EN. Cette opération soustrait l'entrée IN2 de l'entrée IN1 et range le résultat dans la sortie OUT. Si ce résultat est hors de la plage autorisée pour un nombre entier de 16 bits, les bits de débordement et de débordement mémorisé (DEB et DM) sont mis à 1 et la sortie ENO est mise à 0.

Voir aussi Evaluation des bits du mot d'état pour les opérations sur nombres entiers.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	X	X	X	X	X	0	X	X	1

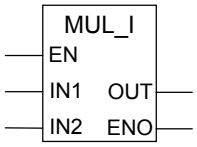
Exemple



La boîte SUB_I est activée si l'état de signal est 1 à l'entrée E 0.0. Le résultat de la soustraction MW0 - MW2 est rangé dans le mot de mémento MW10. Si ce résultat est hors de la plage autorisée pour un nombre entier de 16 bits ou si l'état de signal de l'entrée E 0.0 est égal à 0, la sortie A 4.0 est mise à 0.

7.5 MUL_I : Multiplier entiers de 16 bits

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN1	INT	E, A, M, D, L ou constante	Premier terme de la multiplication
IN2	INT	E, A, M, D, L ou constante	Second terme de la multiplication
OUT	INT	E, A, M, D, L	Résultat de la multiplication
ENO	BOOL	E, A, M, D, L	Sortie de validation

Description

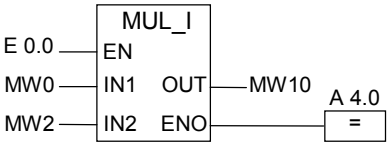
L'opération **Multiplier entiers de 16 bits** est activée si l'état de signal est 1 à l'entrée de validation EN. Cette opération multiplie les entrées IN1 et IN2 et range le résultat dans la sortie OUT. Si ce résultat est hors de la plage autorisée pour un nombre entier de 16 bits, les bits de débordement et de débordement mémorisé (DEB et DM) sont mis à 1 et la sortie ENO est mise à 0.

Voir aussi Evaluation des bits du mot d'état pour les opérations sur nombres entiers.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	X	X	X	X	X	0	X	X	1

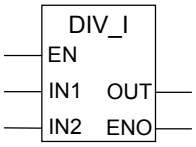
Exemple



La boîte MUL_I est activée si l'état de signal est 1 à l'entrée E 0.0. Le résultat de la multiplication MW0 x MW2 est rangé dans le mot de mémento MW10. Si ce résultat est hors de la plage autorisée pour un nombre entier de 16 bits ou si l'état de signal de l'entrée E 0.0 est égal à 0, la sortie A 4.0 est mise à 0.

7.6 DIV_I : Diviser entiers de 16 bits

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN1	INT	E, A, M, D, L ou constante	Dividende
IN2	INT	E, A, M, D, L ou constante	Diviseur
OUT	INT	E, A, M, D, L	Résultat de la division
ENO	BOOL	E, A, M, D, L	Sortie de validation

Description

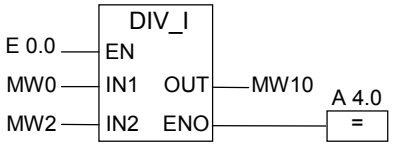
L'opération **Diviser entiers de 16 bits** est activée si l'état de signal est 1 à l'entrée de validation EN. Cette opération divise l'entrée IN1 par l'entrée IN2 et range le quotient (partie entière) dans la sortie OUT. Elle ne fournit pas de reste. Si le quotient est hors de la plage autorisée pour un nombre entier de 16 bits, les bits de débordement et de débordement mémorisé (DEB et DM) sont mis à 1 et la sortie ENO est mise à 0.

Voir aussi Evaluation des bits du mot d'état pour les opérations sur nombres entiers.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	X	X	X	X	X	0	X	X	1

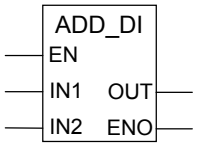
Exemple



La boîte DIV_I est activée si l'état de signal est 1 à l'entrée E 0.0. Le quotient de la division de MW0 par MW2 est rangé dans le mot de mémento MW10. Si ce quotient est hors de la plage autorisée pour un nombre entier de 16 bits ou si l'état de signal de l'entrée E 0.0 est égal à 0, la sortie A 4.0 est mise à 0.

7.7 ADD_DI : Additionner entiers de 32 bits

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN1	DINT	E, A, M, D, L ou constante	Premier terme de l'addition
IN2	DINT	E, A, M, D, L ou constante	Second terme de l'addition
OUT	DINT	E, A, M, D, L	Résultat de l'addition
ENO	BOOL	E, A, M, D, L	Sortie de validation

Description

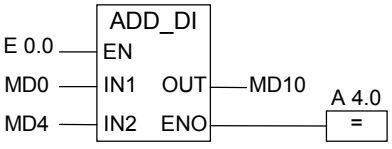
L'opération **Additionner entiers de 32 bits** est activée si l'état de signal est 1 à l'entrée de validation EN. Cette opération additionne les entrées IN1 et IN2 et range le résultat dans la sortie OUT. Si ce résultat est hors de la plage autorisée pour un nombre entier de 32 bits, les bits de débordement et de débordement mémorisé (DEB et DM) sont mis à 1 et la sortie ENO est mise à 0.

Voir aussi Evaluation des bits du mot d'état pour les opérations sur nombres entiers.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	X	X	X	X	X	0	X	X	1

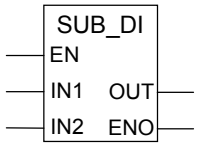
Exemple



La boîte ADD_DI est activée si l'état de signal est 1 à l'entrée de validation EN. Cette opération additionne les entrées IN1 et IN2 et range le résultat dans la sortie OUT. Si ce résultat est hors de la plage autorisée pour un nombre entier de 32 bits, les bits de débordement et de débordement mémorisé (DEB et DM) sont mis à 1 et la sortie ENO est mise à 0.

7.8 SUB_DI : Soustraire entiers de 32 bits

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN1	DINT	E, A, M, D, L ou constante	Valeur dont on soustrait
IN2	DINT	E, A, M, D, L ou constante	Valeur à soustraire
OUT	DINT	E, A, M, D, L	Résultat de la soustraction
ENO	BOOL	E, A, M, D, L	Sortie de validation

Description

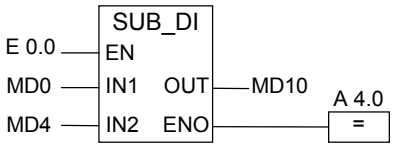
L'opération **Soustraire entiers de 32 bits** est activée si l'état de signal est 1 à l'entrée de validation EN. Cette opération soustrait l'entrée IN2 de l'entrée IN1 et range le résultat dans la sortie OUT. Si ce résultat est hors de la plage autorisée pour un nombre entier de 32 bits, les bits de débordement et de débordement mémorisé (DEB et DM) sont mis à 1 et la sortie ENO est mise à 0.

Voir aussi Evaluation des bits du mot d'état pour les opérations sur nombres entiers.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	X	X	X	X	X	0	X	X	1

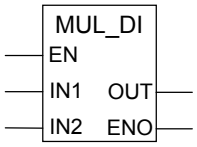
Exemple



La boîte SUB_DI est activée si l'état de signal est 1 à l'entrée E 0.0. Le résultat de la soustraction MD0 - MD4 est rangé dans le double mot de mémento MD10. Si ce résultat est hors de la plage autorisée pour un nombre entier de 32 bits ou si l'état de signal de l'entrée E 0.0 est égal à 0, la sortie A 4.0 est mise à 0 et l'opération n'est pas exécutée.

7.9 MUL_DI : Multiplier entiers de 32 bits

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN1	DINT	E, A, M, D, L ou constante	Premier terme de la multiplication
IN2	DINT	E, A, M, D, L ou constante	Second terme de la multiplication
OUT	DINT	E, A, M, D, L	Résultat de la multiplication
ENO	BOOL	E, A, M, D, L	Sortie de validation

Description

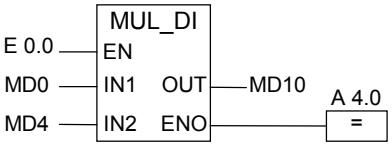
L'opération **Multiplier entiers de 32 bits** est activée si l'état de signal est 1 à l'entrée de validation EN. Cette opération multiplie les entrées IN1 et IN2 et range le résultat dans la sortie OUT. Si ce résultat est hors de la plage autorisée pour un nombre entier de 32 bits, les bits de débordement et de débordement mémorisé (DEB et DM) sont mis à 1 et la sortie ENO est mise à 0.

Voir aussi Evaluation des bits du mot d'état pour les opérations sur nombres entiers.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	X	X	X	X	X	0	X	X	1

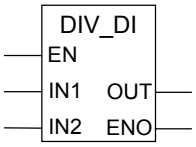
Exemple



La boîte MUL_DI est activée si l'état de signal est 1 à l'entrée E 0.0. Le résultat de la multiplication MD0 x MD4 est rangé dans le double mot de mémento MD10. Si ce résultat est hors de la plage autorisée pour un nombre entier de 32 bits ou si l'état de signal de l'entrée E 0.0 est égal à 0, la sortie A 4.0 est mise à 0.

7.10 DIV_DI : Diviser entiers de 32 bits

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN1	DINT	E, A, M, D, L ou constante	Dividende
IN2	DINT	E, A, M, D, L ou constante	Diviseur
OUT	DINT	E, A, M, D, L	Résultat de la division
ENO	BOOL	E, A, M, D, L	Sortie de validation

Description

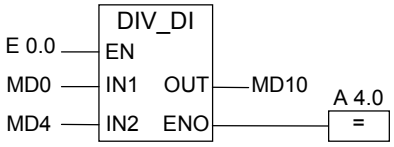
L'opération **Diviser entiers de 32 bits** est activée si l'état de signal est 1 à l'entrée de validation EN. Cette opération divise l'entrée IN1 par l'entrée IN2 et range le quotient (partie entière) dans la sortie OUT. Elle range le quotient sous forme de valeur de 32 bits en format DINT et ne fournit pas de reste. Si le quotient est hors de la plage autorisée pour un nombre entier de 32 bits, les bits de débordement et de débordement mémorisé (DEB et DM) sont mis à 1 et la sortie ENO est mise à 0.

Voir aussi Evaluation des bits du mot d'état pour les opérations sur nombres entiers.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	X	X	X	X	X	0	X	X	1

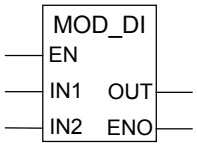
Exemple



La boîte DIV_DI est activée si l'état de signal est 1 à l'entrée E 0.0. Le quotient de la division de MD0 par MD4 est rangé dans le double mot de mémento MD10. Si ce quotient est hors de la plage autorisée pour un nombre entier de 32 bits ou si l'état de signal de l'entrée E 0.0 est égal à 0, la sortie A 4.0 est mise à 0.

7.11 MOD_DI : Reste de division (32 bits)

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN1	DINT	E, A, M, D, L ou constante	Dividende
IN2	DINT	E, A, M, D, L ou constante	Diviseur
OUT	DINT	E, A, M, D, L	Reste
ENO	BOOL	E, A, M, D, L	Sortie de validation

Description

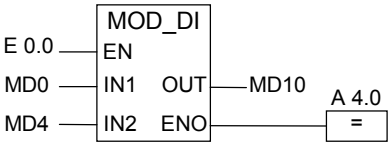
L'opération **Reste de division (32 bits)** est activée si l'état de signal est 1 à l'entrée de validation EN. Cette opération divise l'entrée IN1 par l'entrée IN2 et range le reste dans la sortie OUT. Si ce reste est hors de la plage autorisée pour un nombre entier de 32 bits, les bits de débordement et de débordement mémorisé (DEB et DM) sont mis à 1 et la sortie ENO est mise à 0.

Voir aussi Evaluation des bits du mot d'état pour les opérations sur nombres entiers.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	X	X	X	X	X	0	X	X	1

Exemple



La boîte MOD_DI est activée si l'état de signal est 1 à l'entrée E 0.0. Le reste de la division de MD0 par MD4 est rangé dans le double mot de mémento MD10. Si ce reste est hors de la plage autorisée pour un nombre entier de 32 bits ou si l'état de signal de l'entrée E 0.0 est égal à 0, la sortie A 4.0 est mise à 0.

8 Opérations arithmétiques sur nombres réels

8.1 Vue d'ensemble des opérations arithmétiques sur nombres réels

Description

Les nombres à virgule flottante IEEE de 32 bits ont le type de données REAL. Les opérations arithmétiques sur nombres à virgule flottante permettent d'exécuter les fonctions arithmétiques suivantes sur **deux** nombres réels IEEE de 32 bits :

- ADD_R : Additionner nombres réels
- SUB_R : Soustraire nombres réels
- MUL_R : Multiplier nombres réels
- DIV_R : Diviser nombres réels

Les opérations arithmétiques sur nombres à virgule flottante permettent d'exécuter les fonctions arithmétiques suivantes sur **un** nombre réels IEEE de 32 bits :

- valeur absolue (ABS) d'un nombre réel,
- carré (SQR) et racine carrée (SQRT) d'un nombre réel,
- logarithme naturel (LN) d'un nombre réel,
- valeur exponentielle (EXP) de base e (= 2,71828...) d'un nombre réel,
- fonctions trigonométriques d'angles représentés sous forme de nombres à virgule flottante IEEE de 32 bits :
  - sinus d'un nombre réel (SIN) et arc sinus d'un nombre réel (ASIN),
  - cosinus d'un nombre réel (COS) et arc cosinus d'un nombre réel (ACOS),
  - tangente d'un nombre réel (TAN) et arc tangente d'un nombre réel (ATAN).

Voir aussi Evaluation des bits du mot d'état dans les opérations sur nombres à virgule flottante

8.2 Evaluation des bits du mot d'état pour les opérations sur nombres réels

Description

Les opérations arithmétiques sur nombres réels affectent les bits suivants du mot d'état :

- BI0 et BI1,
- DEB
- et DM.

Les tableaux ci-dessous montrent l'état de signal des bits du mot d'état pour les résultats d'opérations sur nombres à virgule flottante (32 bits) :

Plage autorisée	**BI1**	**BI0**	**DEB**	**DM**
+0, -0 (zéro)	0	0	0	*
-3.402823E+38 < résultat < -1.175494E-38 (nombre négatif)	0	1	0	*
+1.175494E-38 < résultat < 3.402824E+38 (nombre positif)	1	0	0	*

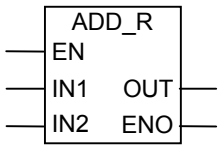
* Le bit DM n'est pas influencé par le résultat de l'opération.

Plage non autorisée	**BI1**	**BI0**	**DEB**	**DM**
Dépassement bas -1.175494E-38 < résultat < - 1.401298E-45 (nombre négatif)	0	0	1	1
Dépassement bas +1.401298E-45 < résultat < +1.175494E-38 (nombre positif)	0	0	1	1
Débordement résultat < -3.402823E+38 (nombre négatif)	0	1	1	1
Débordement résultat > 3.402823E+38 (nombre positif)	1	0	1	1
Pas un nombre réel correct ou opération illicite (valeur d'entrée hors de la plage de valeurs autorisée)	1	1	1	1

8.3 Opérations de base

8.3.1 ADD_R : Additionner nombres réels

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN1	REAL	E, A, M, D, L ou constante	Premier terme de l'addition
IN2	REAL	E, A, M, D, L ou constante	Second terme de l'addition
OUT	REAL	E, A, M, D, L	Résultat de l'addition
ENO	BOOL	E, A, M, D, L	Sortie de validation

Description

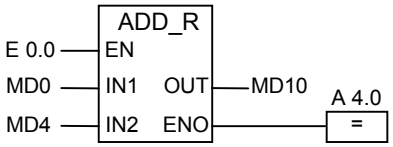
L'opération **Additionner nombres réels** est activée si l'état de signal est 1 à l'entrée de validation EN. Cette opération additionne les entrées IN1 et IN2 et range le résultat dans la sortie OUT. Si l'une des entrées ou le résultat n'est pas un nombre réel, les bits de débordement et de débordement mémorisé (DEB et DM) sont mis à 1 et la sortie ENO est mise à 0.

Vous trouverez des informations sur l'évaluation des bits du mot d'état à la rubrique Evaluation des bits du mot d'état pour les opérations sur nombres réels.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	X	X	X	X	X	0	X	X	1

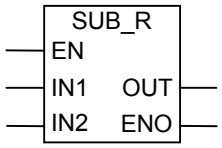
Exemple



La boîte ADD_R est activée si l'état de signal est 1 à l'entrée E 0.0. Le résultat de l'addition MD0 + MD4 est rangé dans le double mot de mémento MD10. Si l'une des entrées ou le résultat n'est pas un nombre réel ou si l'état de signal de l'entrée E 0.0 est égal à 0, la sortie A 4.0 est mise à 0.

8.3.2 SUB_R : Soustraire nombres réels

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN1	REAL	E, A, M, D, L ou constante	Valeur dont on soustrait
IN2	REAL	E, A, M, D, L ou constante	Valeur à soustraire
OUT	REAL	E, A, M, D, L	Résultat de la soustraction
ENO	BOOL	E, A, M, D, L	Sortie de validation

Description

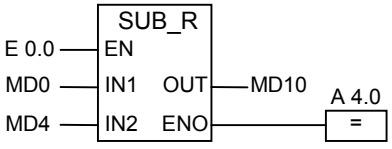
L'opération **Soustraire nombres réels** est activée si l'état de signal est 1 à l'entrée de validation EN. Cette opération soustrait l'entrée IN2 de l'entrée IN1 et range le résultat dans la sortie OUT. Si l'une des entrées ou le résultat n'est pas un nombre réel, les bits de débordement et de débordement mémorisé (DEB et DM) sont mis à 1 et la sortie ENO est mise à 0.

Vous trouverez des informations sur l'évaluation des bits du mot d'état à la rubrique Evaluation des bits du mot d'état pour les opérations sur nombres réels.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	X	X	X	X	X	0	X	X	1

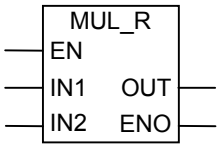
Exemple



La boîte SUB_R est activée si l'état de signal est 1 à l'entrée E 0.0. Le résultat de la soustraction MD0 - MD4 est rangé dans le double mot de mémento MD10. Si l'une des entrées ou le résultat n'est pas un nombre réel ou si l'état de signal de l'entrée E 0.0 est égal à 0, la sortie A 4.0 est mise à 0.

8.3.3 MUL_R : Multiplier nombres réels

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN1	REAL	E, A, M, D, L ou constante	Premier terme de la multiplication
IN2	REAL	E, A, M, D, L ou constante	Second terme de la multiplication
OUT	REAL	E, A, M, D, L	Résultat de la multiplication
ENO	BOOL	E, A, M, D, L	Sortie de validation

Description

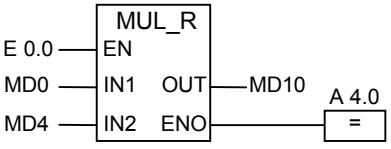
L'opération **Multiplier nombres réels** est activée si l'état de signal est 1 à l'entrée de validation EN. Cette opération multiplie les entrées IN1 et IN2 et range le résultat dans la sortie OUT. Si l'une des entrées ou le résultat n'est pas un nombre réel, les bits de débordement et de débordement mémorisé (DEB et DM) sont mis à 1 et la sortie ENO est mise à 0.

Vous trouverez des informations sur l'évaluation des bits du mot d'état à la rubrique Evaluation des bits du mot d'état pour les opérations sur nombres réels.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	X	X	X	X	X	0	X	X	1

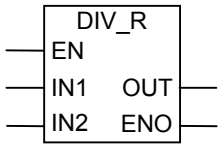
Exemple



La boîte MUL_R est activée si l'état de signal est 1 à l'entrée E 0.0. Le résultat de la multiplication MD0 x MD4 est rangé dans le double mot de mémento MD10. Si l'une des entrées ou le résultat n'est pas un nombre réel ou si l'état de signal de l'entrée E 0.0 est égal à 0, la sortie A 4.0 est mise à 0.

8.3.4 DIV_R : Diviser nombres réels

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN1	REAL	E, A, M, D, L ou constante	Dividende
IN2	REAL	E, A, M, D, L ou constante	Diviseur
OUT	REAL	E, A, M, D, L	Résultat de la division
ENO	BOOL	E, A, M, D, L	Sortie de validation

Description

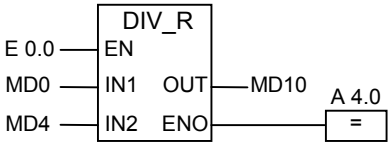
L'opération **Diviser nombres réels** est activée si l'état de signal est 1 à l'entrée de validation EN. Cette opération divise l'entrée IN1 par l'entrée IN2 et range le résultat dans la sortie OUT. Si l'une des entrées ou le résultat n'est pas un nombre réel, les bits de débordement et de débordement mémorisé (DEB et DM) sont mis à 1 et la sortie ENO est mise à 0.

Vous trouverez des informations sur l'évaluation des bits du mot d'état à la rubrique Evaluation des bits du mot d'état pour les opérations sur nombres réels.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	X	X	X	X	X	0	X	X	1

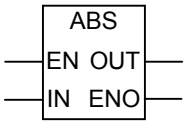
Exemple



La boîte DIV_R est activée si l'état de signal est 1 à l'entrée E 0.0. Le résultat de la division de MD0 par MD4 est rangé dans le double mot de mémento MD10. Si l'une des entrées ou le résultat n'est pas un nombre réel ou si l'état de signal de l'entrée E 0.0 est égal à 0, la sortie A 4.0 est mise à 0.

8.3.5 ABS : Valeur absolue d'un nombre réel

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN	REAL	E, A, M, D, L ou constante	Valeur d'entrée : nombre réel
OUT	REAL	E, A, M, D, L	Valeur de sortie : valeur absolue du nombre réel
ENO	BOOL	E, A, M, D, L	Sortie de validation

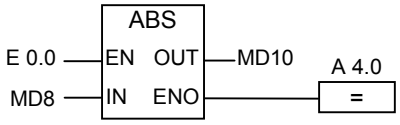
Description

L'opération **Valeur absolue d'un nombre réel** permet d'obtenir la valeur absolue d'un nombre réel.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	X	-	-	-	-	0	X	X	1

Exemple



Si E 0.0 est à 1, la valeur absolue du nombre réel figurant dans MD8 est transmise dans MD12.

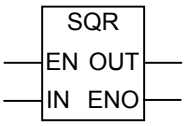
MD8 = +6,234 donne MD12 = 6,234.

La sortie A 4.0 est mise à 0 si la conversion n'est pas exécutée (ENO = EN = 0).

8.4 Opérations étendues

8.4.1 SQR : Carré d'un nombre réel

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN	REAL	E, A, M, D, L ou constante	Nombre réel
OUT	REAL	E, A, M, D, L	Carré du nombre réel
ENO	BOOL	E, A, M, D, L	Sortie de validation

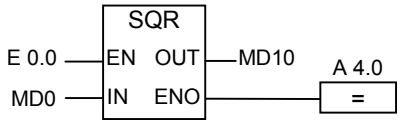
Description

L'opération **Carré d'un nombre réel** calcule le carré d'un nombre réel. Si l'entrée ou le résultat n'est pas un nombre réel, les bits de débordement et de débordement mémorisé (DEB et DM) sont mis à 1 et la sortie ENO est mise à 0.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	X	X	X	X	X	0	X	X	1

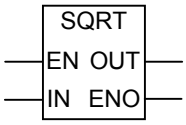
Exemple



La boîte SQR est activée si l'état de signal est 1 à l'entrée E 0.0. Le résultat de la mise au carré de MD0 est rangé dans le double mot de mémento MD10. Si MD0 est inférieur à 0, si l'entrée ou le résultat n'est pas un nombre réel ou si l'état de signal de l'entrée E 0.0 est égal à 0, la sortie A 4.0 est mise à 0.

8.4.2 SQRT : Racine carrée d'un nombre réel

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN	REAL	E, A, M, D, L ou constante	Nombre réel
OUT	REAL	E, A, M, D, L	Racine carrée du nombre réel
ENO	BOOL	E, A, M, D, L	Sortie de validation

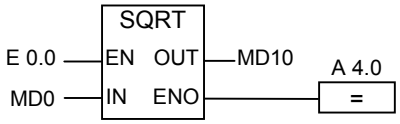
Description

L'opération **Racine carrée d'un nombre réel** calcule la racine carrée d'un nombre réel. Cette opération délivre un résultat positif si l'opérande est supérieur à zéro. Si l'entrée ou le résultat n'est pas un nombre réel, les bits de débordement et de débordement mémorisé (DEB et DM) sont mis à 1 et la sortie ENO est mise à 0.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	X	X	X	X	X	0	X	X	1

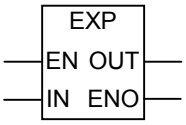
Exemple



La boîte SQRT est activée si l'état de signal est 1 à l'entrée E 0.0. Le résultat de l'extraction de racine carrée de MD0 est rangé dans le double mot de mémento MD10. Si MD0 est inférieur à 0, si l'entrée ou le résultat n'est pas un nombre réel ou si l'état de signal de l'entrée E 0.0 est égal à 0, la sortie A 4.0 est mise à 0.

8.4.3 EXP : Valeur exponentielle d'un nombre réel

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN	REAL	E, A, M, D, L ou constante	Nombre réel
OUT	REAL	E, A, M, D, L	Valeur exponentielle du nombre réel
ENO	BOOL	E, A, M, D, L	Sortie de validation

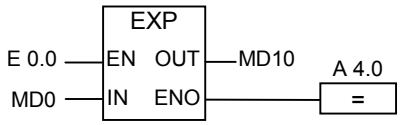
Description

L'opération **Valeur exponentielle d'un nombre réel** calcule la valeur exponentielle de base e (= 2,71828...) d'un nombre réel. Si l'entrée ou le résultat n'est pas un nombre réel, les bits de débordement et de débordement mémorisé (DEB et DM) sont mis à 1 et la sortie ENO est mise à 0.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	X	X	X	X	X	0	X	X	1

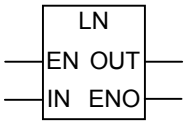
Exemple



La boîte EXP est activée si l'état de signal est 1 à l'entrée E 0.0. Le résultat du calcul de la valeur exponentielle de MD0 est rangé dans le double mot de mémento MD10. Si l'entrée ou le résultat n'est pas un nombre réel ou si l'état de signal de l'entrée E 0.0 est égal à 0, la sortie A 4.0 est mise à 0.

8.4.4 LN : Logarithme naturel d'un nombre réel

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN	REAL	E, A, M, D, L ou constante	Nombre réel
OUT	REAL	E, A, M, D, L	Logarithme naturel du nombre réel
ENO	BOOL	E, A, M, D, L	Sortie de validation

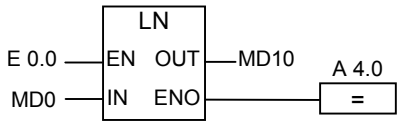
Description

L'opération **Logarithme naturel d'un nombre réel** calcule le logarithme naturel d'un nombre réel. Si l'entrée ou le résultat n'est pas un nombre réel, les bits de débordement et de débordement mémorisé (DEB et DM) sont mis à 1 et la sortie ENO est mise à 0.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	X	X	X	X	X	0	X	X	1

Exemple



La boîte LN est activée si l'état de signal est 1 à l'entrée E 0.0. Le résultat du calcul du logarithme naturel de MD0 est rangé dans le double mot de mémento MD10. Si MD0 est inférieur à 0, si l'entrée ou le résultat n'est pas un nombre réel ou si l'état de signal de l'entrée E 0.0 est égal à 0, la sortie A 4.0 est mise à 0.

8.4.5 Fonctions trigonométriques d'angles sous forme de nombres réels

Description

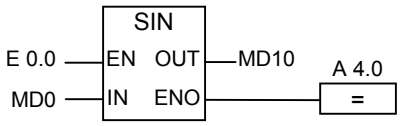
Les opérations suivantes vous permettent d'effectuer des fonctions trigonométriques d'angles représentés sous forme de nombres à virgule flottante IEEE de 32 bits :

Opération	Signification
SIN	Sinus d'un nombre réel d'un angle exprimé en radians
ASIN	Arc sinus d'un nombre réel. Le résultat de l'opération est un angle exprimé en radians. La valeur de cet angle est comprise dans la plage suivante : $-\pi / 2 <=$ arc sinus $<= + \pi / 2$, avec $\pi = 3.14...$
COS	Cosinus d'un nombre réel d'un angle exprimé en radians
ACOS	Arc cosinus d'un nombre réel. Le résultat de l'opération est un angle exprimé en radians. La valeur de cet angle est comprise dans la plage suivante : $0 <=$ arc cosinus $<= + \pi$, avec $\pi = 3.14...$
TAN	Tangente d'un nombre réel d'un angle exprimé en radians
ATAN	Arc tangente d'un nombre réel. Le résultat de l'opération est un angle exprimé en radians. La valeur de cet angle est comprise dans la plage suivante : $- \pi / 2 <=$ arc tangente $<= + \pi / 2$, avec $\pi = 3.14...$

Mot d'état

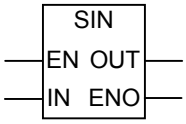
	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	x	x	x	x	x	0	x	x	1

Exemple



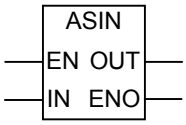
La boîte SIN est activée si l'état de signal est 1 à l'entrée E 0.0. Le résultat du calcul du sinus de MD0 est rangé dans le double mot de mémento MD10. Si l'entrée ou le résultat n'est pas un nombre réel ou si l'état de signal de l'entrée E 0.0 est égal à 0, la sortie A 4.0 est mise à 0.

Représentation



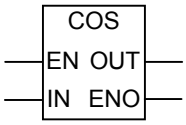
Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN	REAL	E, A, M, D, L ou constante	Nombre réel
OUT	REAL	E, A, M, D, L	Sinus du nombre réel
ENO	BOOL	E, A, M, D, L	Sortie de validation

Représentation



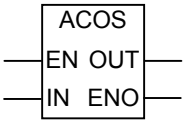
Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN	REAL	E, A, M, D, L ou constante	Nombre réel
OUT	REAL	E, A, M, D, L	Arc sinus du nombre réel
ENO	BOOL	E, A, M, D, L	Sortie de validation

Représentation



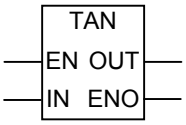
Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN	REAL	E, A, M, D, L ou constante	Nombre réel
OUT	REAL	E, A, M, D, L	Cosinus du nombre réel
ENO	BOOL	E, A, M, D, L	Sortie de validation

Représentation



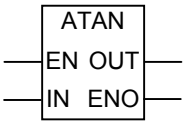
Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN	REAL	E, A, M, D, L ou constante	Nombre réel
OUT	REAL	E, A, M, D, L	Arc cosinus du nombre réel
ENO	BOOL	E, A, M, D, L	Sortie de validation

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN	REAL	E, A, M, D, L ou constante	Nombre réel
OUT	REAL	E, A, M, D, L	Tangente du nombre réel
ENO	BOOL	E, A, M, D, L	Sortie de validation

Représentation

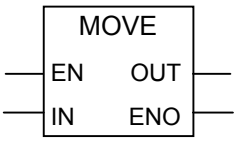


Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN	REAL	E, A, M, D, L ou constante	Nombre réel
OUT	REAL	E, A, M, D, L	Arc tangente du nombre réel
ENO	BOOL	E, A, M, D, L	Sortie de validation

9 Opérations de transfert

9.1 MOVE : Affecter valeur

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN	Tous types de données simples de 8, 16 ou 32 bits	E, A, M, D, L ou constante	Valeur source
OUT	Tous types de données élémentaires de 8, 16 ou 32 bits	E, A, M, D, L	Adresse de destination
ENO	BOOL	E, A, M, D, L	Sortie de validation

Description

L'opération **Affecter valeur** permet d'initialiser des variables avec des valeurs précises.

La valeur indiquée à l'entrée IN est copiée dans l'opérande précisé dans la sortie OUT. L'état de signal de ENO est identique à celui de EN.

La boîte MOVE permet de copier tous les types de données élémentaires de 8, 16 ou 32 bits. Pour copier des types de données utilisateur tels que des tableaux ou des structures, vous devez faire appel à la fonction système SFC20 "BLKMOV".

Le relais de masquage MCR influence l'opération **Affecter valeur**.

Mot d'état

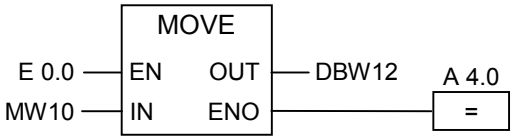
	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	1	-	-	-	-	0	1	1	1

Nota

Lors de l'affectation d'une valeur à un type de données de longueur différente, les octets de poids fort sont, le cas échéant, tronqués ou complétés par des zéro.

Exemple : Double mot	**1111 1111**	**0000 1111**	**1111 0000**	**0101 0101**
Affectation	**Résultat**			
à un double mot :	1111 1111	0000 1111	1111 0000	0101 0101
à un octet :				0101 0101
à un mot :			1111 0000	0101 0101
Exemple : Octet				**1111 0000**
Affectation	**Résultat**			
à un octet :				1111 0000
à un mot :			0000 0000	1111 0000
à un double mot :	0000 0000	0000 0000	0000 0000	1111 0000

Exemple



L'opération est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le contenu de MW10 est copié dans le mot de données 12 du bloc de données en cours.

A 4.0 est mis à 1 si l'opération est exécutée.

10 Opérations de gestion d'exécution de programme

10.1 Vue d'ensemble des opérations de gestion d´exécution de programme

Description

Vous disposez des opérations de gestion d´exécution de programme suivantes :

- CALL : Appeler FC/SFC sans paramètre
- CALL_FB : Appeler FB
- CALL_FC : Appeler FC
- CALL_SFB : Appeler SFB
- CALL_SFC : Appeler SFC
- Appeler multi-instances
- Appeler un bloc dans une bibliothèque

- Fonctions du relais de masquage
- Remarques importantes sur l'utilisation de la fonctionnalité MCR
- MCR< : Relais de masquage en fonction
- MCR> : Relais de masquage hors fonction
- MCRA : Activer relais de masquage
- MCRD : Désactiver relais de masquage

- RET : Retour

10.2 CALL : Appeler FC/SFC sans paramètre

Représentation

<FC/SFC numéro>

Paramètre	Type de données	Zone de mémoire	Description
Numéro	BLOCK_FC	-	Numéro de la FC ou de la SFC (par exemple, FC10 ou SFC59). Les SFC disponibles dépendent de votre CPU. Un appel conditionnel avec un paramètre de type de données BLOCK_FC comme opérande n'est possible que dans un FB et pas dans une FC.

Description

L'opération **Appeler FC/SFC sans paramètre** permet d'appeler une fonction (FC) ou une fonction système (SFC) qui n'a pas de paramètre. L'appel est inconditionnel ou conditionnel selon la combinaison précédente (voir exemple).

Dans la section d'instructions d'une fonction (FC), vous ne pouvez pas indiquer de paramètre de type de données BLOCK_FC comme opérande lors d'un appel conditionnel. En revanche, vous pouvez spécifier un paramètre de type BLOCK_FC comme opérande dans un bloc fonctionnel (FB).

Un appel conditionnel n'est exécuté que si le résultat logique est à 1. Si un appel conditionnel n'est pas exécuté, le RLG est égal à 0 après l'opération d'appel. Si l'opération a lieu, elle fonctionne comme suit :

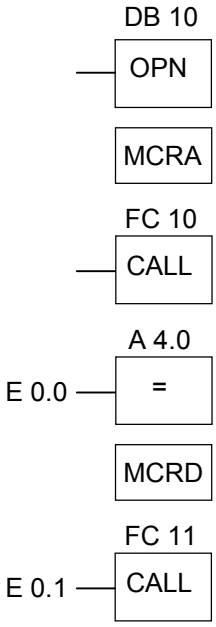
- Elle sauvegarde l'adresse de retour au bloc appelant.
- Elle sauvegarde les deux registres de bloc de données (bloc de données et bloc de données d'instance).
- Elle change la zone de données locales en cours en zone de données locales précédente.
- Elle empile le bit MA (bit MCR actif) dans la pile des blocs.
- Elle crée la zone de données locales en cours pour la fonction ou la fonction système appelée.

Ensuite, le programme poursuit le traitement dans le bloc appelé.

Mot d'état

		RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Conditionnel	Ecriture	-	-	-	-	0	0	1	1	0
Inconditionnel	Ecriture	-	-	-	-	0	0	1	-	0

Exemple



Si l'appel inconditionnel de la FC10 est exécuté, l'opération CALL fonctionne comme suit :

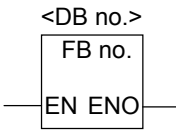
- Elle sauvegarde l'adresse de retour au FB en cours.
- Elle sauvegarde les sélecteurs pour le DB10 et pour le bloc de données d'instance du FB.
- Elle empile le bit MA, mis à 1 par l'opération MCRA, dans la pile des blocs et remet ce bit à 0 pour la fonction FC10 appelée.

Le traitement du programme se poursuit dans la FC10. Si vous voulez utiliser la fonction MCR dans la FC10, vous devez l'y réactiver. A la fin de la FC10, le traitement du programme revient au FB appelant. Le bit MA est restauré. Le DB10 et le bloc de données d'instance du FB utilisateur redeviennent les DB en cours, quels qu'aient été les DB utilisés par la FC10.

Au retour de la FC10, l'état de signal en E 0.0 est affecté à la sortie A 4.0. L'appel de la FC11 étant conditionnel, il n'est exécuté que si l'état de signal en E 0.1 est 1. S'il est exécuté, il fonctionne comme l'appel de la FC10.

10.3 CALL_FB : Appeler FB

Représentation



La représentation dépend du bloc fonctionnel (à savoir si des paramètres sont présents et combien). L'entrée EN, la sortie ENO et le nom ou le numéro du FB doivent être présents.

Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, L, D	Entrée de validation
ENO	BOOL	E, A, M, L, D	Sortie de validation
FB no. DB no.	BLOCK_FB BLOCK_DB	- -	Numéros du FB et du DB. La plage dépend de la CPU.

Description de l'opération

L'opération **Appeler FB** (CALL_FB) permet d'appeler un bloc fonctionnel (FB). L'appel n'est exécuté que si EN = 1. Si l'opération a lieu, elle fonctionne comme suit :

- Elle sauvegarde l'adresse de retour au bloc appelant.
- Elle sauvegarde les deux registres de bloc de données (bloc de données et bloc de données d'instance).
- Elle change la zone de données locales en cours en zone de données locales précédente.
- Elle empile le bit MA (bit MCR actif) dans la pile des blocs.
- Elle crée la nouvelle zone de données locales pour le bloc fonctionnel appelé.

Mot d'état

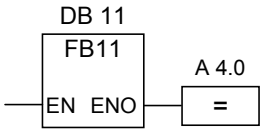
		RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Conditionnel	Ecriture	X	-	-	-	0	0	X	X	X
Inconditionnel	Ecriture	-	-	-	-	0	0	X	X	X

Exemple

Réseau 1

Réseau 2

Réseau 3



Réseau 4

Les réseaux CONT représentés ci-dessus sont des parties de programme d'un bloc fonctionnel utilisateur. Ce bloc fonctionnel ouvre le DB10 et active la fonction MCR. Si l'appel inconditionnel du FB11 est exécuté, voici ce qui se passe :

L'adresse de retour au bloc fonctionnel appelant et les sélecteurs pour le DB10 et pour le bloc de données d'instance de ce bloc fonctionnel sont sauvegardés. Le bit MA qui a été mis à 1 par l'opération MCRA est empilé dans la pile des blocs, puis mis à 0 pour le bloc FB11 appelé. Ensuite, le traitement du programme se poursuit dans le bloc FB11. Si vous voulez utiliser la fonction MCR dans le FB11, vous devez l'y réactiver. Il faut sauvegarder l'état du RLG dans le bit RB via l'opération [SAVE] afin de pouvoir procéder à une évaluation des erreurs dans le FB appelant. A la fin du FB11, le traitement du programme revient au FB appelant. Le bit MA est restauré et le bloc de données d'instance du bloc fonctionnel utilisateur redevient le DB d'instance en cours. Si le FB11 est exécuté sans erreur, ENO et donc A 4.0 sont à 1.

Nota

Pour des appels de FB ou de SFB, le numéro du bloc de données ouvert précédemment est perdu. Il faut donc à nouveau ouvrir le DB requis.

10.4 CALL_FC : Appeler FC

Représentation

La représentation dépend de la fonction (à savoir si des paramètres sont présents et combien). L'entrée EN, la sortie ENO et le nom ou le numéro de la FC doivent être présents.

Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, L, D	Entrée de validation
ENO	BOOL	E, A, M, L, D	Sortie de validation
FC no.	BLOCK_FC	-	Numéro de la FC. La plage dépend de la CPU

Description

L'opération **Appeler FC** (CALL_FC) permet d'appeler une fonction (FC). L'appel n'est exécuté que si EN = 1. Si l'opération a lieu, elle fonctionne comme suit :

- Elle sauvegarde l'adresse de retour au bloc appelant.
- Elle change la zone de données locales en cours en zone de données locales précédente.
- Elle empile le bit MA (bit MCR actif) dans la pile des blocs.
- Elle crée la zone de données locales en cours pour la fonction appelée.

Ensuite, le programme poursuit le traitement dans le bloc appelé.

Pour déterminer la sortie de validation ENO, le bit RB est interrogé ; l'état de signal souhaité (évaluation d'erreurs) doit lui être affecté par l'utilisateur dans le bloc appelé, à l'aide de l'opération [SAVE].

Lorsque vous appelez une fonction (FC) et que la table de déclaration des variables du bloc appelé comporte des déclarations du type IN, OUT et IN_OUT, ces variables s'affichent sous forme de liste de paramètres formels dans le programme du bloc appelant.

Lors de l'appel des FC, **vous devez impérativement** affecter des paramètres effectifs aux paramètres formels à l'endroit de l'appel. D'éventuelles valeurs initiales dans la déclaration de la FC sont insignifiantes.

Mot d'état

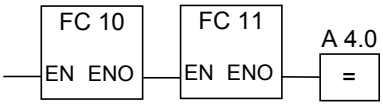
		RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Conditionnel	Ecriture	X	-	-	-	0	0	X	X	X
Inconditionnel	Ecriture	-	-	-	-	0	0	X	X	X

Exemple

Réseau 1

Réseau 2

Réseau 3



Les réseaux CONT représentés ci-dessus sont des parties de programme d'un bloc fonctionnel utilisateur. Ce bloc fonctionnel ouvre le DB10 et active la fonction MCR. Si l'appel inconditionnel de la FC10 est exécuté, voici ce qui se passe :

L'opération sauvegarde l'adresse de retour au FB en cours et les sélecteurs pour le DB10 et pour le bloc de données d'instance du FB. Elle empile le bit MA, mis à 1 par l'opération MCRA, dans la pile des blocs et remet ce bit à 0 pour la fonction FC10 appelée. Le traitement du programme se poursuit dans la FC10. Si vous voulez utiliser la fonction MCR dans la FC10, vous devez l'y réactiver. Il faut sauvegarder l'état du RLG dans le bit RB à l'aide de l'opération [SAVE] afin de pouvoir procéder à une évaluation des erreurs dans le FB appelant. A la fin de la FC10, le traitement du programme revient au FB appelant. Le bit MA est restauré. Le programme se poursuit dans le FB appelant selon l'état de signal de la sortie de validation ENO :

ENO = 1 Traitement de la FC11

ENO = 0 Le traitement commence dans le réseau suivant.

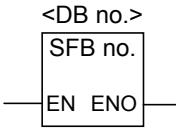
Si la FC11 est également exécutée sans erreur, ENO et donc A 4.0 sont à 1.

Nota

Après retour au bloc appelant, il n'est pas toujours certain que le DB ouvert précédemment soit de nouveau ouvert. Veuillez observer les informations dans le fichier LISEZMOI.

10.5 CALL_SFB : Appeler SFB

Représentation



La représentation dépend du bloc fonctionnel système (à savoir si des paramètres sont présents et combien). L'entrée EN, la sortie ENO et le nom ou le numéro du SFB doivent être présents.

Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, L, D	Entrée de validation
ENO	BOOL	E, A, M, L, D	Sortie de validation
SFB no. DB no.	BLOCK_SFB BLOCK_DB	- -	Numéro du SFB/DB. La plage dépend de la CPU.

Description de l'opération

L'opération **Appeler SFB** (CALL_SFB) permet d'appeler un bloc fonctionnel système (SFB). L'appel n'est exécuté que si EN = 1. Si l'opération a lieu, elle fonctionne comme suit :

- Elle sauvegarde l'adresse de retour au bloc appelant.
- Elle sauvegarde les deux registres de bloc de données (bloc de données et bloc de données d'instance).
- Elle change la zone de données locales en cours en zone de données locales précédente.
- Elle empile le bit MA (bit MCR actif) dans la pile des blocs.
- Elle crée la nouvelle zone de données locales pour le bloc fonctionnel système appelé.

Ensuite, le traitement du programme se poursuit dans le bloc fonctionnel système appelé. ENO est à 1 si le SFB a été appelé (EN = 1) et si aucune erreur n'est apparue.

Mot d'état

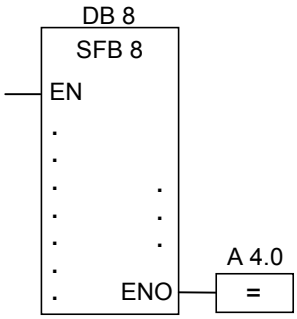
		RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Conditionnel	Ecriture	X	-	-	-	0	0	X	X	X
Inconditionnel	Ecriture	-	-	-	-	0	0	X	X	X

Exemple

Réseau 1

Réseau 2

Réseau 3



Réseau 4

Les opérations CONT représentées ci-dessus sont des parties de programme d'un bloc fonctionnel utilisateur. Ce bloc fonctionnel ouvre le DB10 et active la fonction MCR. Si l'appel inconditionnel du bloc SFB8 est exécuté, voici ce qui se passe :

L'adresse de retour au bloc fonctionnel appelant et les sélecteurs pour le DB10 et pour le bloc de données d'instance de ce bloc fonctionnel sont sauvegardés. Le bit MA qui a été mis à 1 par l'opération MCRA est empilé dans la pile des blocs, puis mis à 0 pour le bloc SFB8 appelé. Ensuite, le traitement du programme se poursuit dans le bloc fonctionnel SFB8. A la fin du SFB8, le traitement du programme revient au FB appelant. Le bit MA est restauré et le bloc de données d'instance du bloc fonctionnel utilisateur redevient le DI en cours. Si le SFB8 est exécuté sans erreur, ENO et donc A 4.0 sont à 1.

Nota

Pour des appels de FB ou de SFB, le numéro du bloc de données ouvert précédemment est perdu. Le DB requis doit être de nouveau ouvert.

10.6 CALL_SFC : Appeler SFC

Représentation

La représentation dépend de la fonction système (à savoir si des paramètres sont présents et combien). L'entrée EN, la sortie ENO et le nom ou le numéro de la SFC doivent être présents.

Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, L, D	Entrée de validation
ENO	BOOL	E, A, M, L, D	Sortie de validation
SFC no.	BLOCK_SFC	-	Numéro de la SFC. La plage dépend de la CPU.

Description de l'opération

L'opération **Appeler SFC** (CALL_SFC) permet d'appeler une fonction système (SFC). L'appel n'est exécuté que si EN = 1. Si l'opération a lieu, elle fonctionne comme suit :

- Elle sauvegarde l'adresse de retour au bloc appelant.
- Elle change la zone de données locales en cours en zone de données locales précédente.
- Elle empile le bit MA (bit MCR actif) dans la pile des blocs.
- Elle crée la nouvelle zone de données locales pour la fonction appelée.

Ensuite, le programme poursuit le traitement dans le bloc appelé. ENO est à 1 si la fonction a été appelée (EN est à 1) et si aucune erreur n'est apparue.

Mot d'état

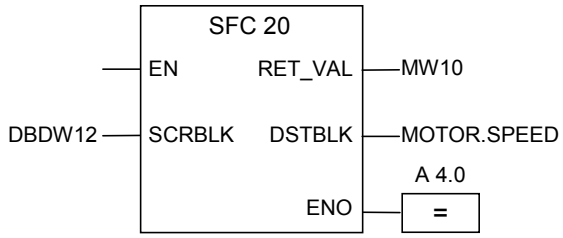
		RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Conditionnel	Ecriture	X	-	-	-	0	0	X	X	X
Inconditionnel	Ecriture	-	-	-	-	0	0	X	X	X

Exemple

Réseau 1

Réseau 2

Réseau 3



Les réseaux CONT représentés ci-dessus sont des parties de programme d'un bloc fonctionnel utilisateur. Ce bloc fonctionnel ouvre le DB10 et active la fonction MCR. Si l'appel inconditionnel de la SFC20 est exécuté, voici ce qui se passe :

L'adresse de retour au bloc fonctionnel appelant et les sélecteurs pour le DB10 et pour le bloc de données d'instance de ce bloc fonctionnel sont sauvegardés. Le bit MA qui a été mis à 1 par l'opération MCRA est empilé dans la pile des blocs, puis mis à 0 pour la fonction SFC20. Ensuite, le traitement du programme se poursuit dans la fonction SFC20. A la fin de la SFC20, le traitement du programme revient au FB appelant. Le bit MA est restauré.

Le programme se poursuit dans le FB appelant selon l'état de signal de la sortie de validation ENO :

ENO = 1 A 4.0 = 1

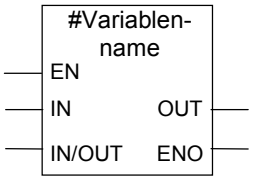
ENO = 0 A 4.0 = 0

Nota

Après retour au bloc appelant, il n'est pas toujours certain que le DB ouvert précédemment soit de nouveau ouvert. Veuillez observer les informations dans le fichier LISEZMOI.

10.7 Appeler multi-instances

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
ENO	BOOL	E, A, M, D, L	Sortie de validation
# nom-variable	FB/SFB	-	Nom de la multi-instance

Description

Vous créez une multi-instance par la déclaration d'une variable statique de type de données "bloc fonctionnel". Seules les multi-instances déjà déclarées apparaissent dans le catalogue des éléments de programme. La représentation d'une multi-instance dépend de l'existence de paramètres et de leur nombre. L'entrée EN, la sortie ENO et le nom de la variable sont toujours présents.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	-	-	-	0	0	X	X	X

10.8 Appeler un bloc dans une bibliothèque

La liste des bibliothèques connues dans le gestionnaire de projets SIMATIC vous est proposée.

Vous pouvez y choisir des blocs

- qui sont intégrés dans le système d'exploitation de votre CPU (bibliothèque "Standard Library" pour les projets STEP 7 de la version 3 et bibliothèque "stdlibs (V2)" pour les projets STEP 7 de la version 2) ;
- que vous avez rangés vous-même dans des bibliothèques parce que vous avez l'intention de les utiliser plusieurs fois.

10.9 Fonctions du relais de masquage

Remarques importantes sur l'utilisation de la fonctionnalité MCR

Définition du relais de masquage

Le relais de masquage MCR (en anglais Master Control Relay, voir aussi la rubrique Relais de masquage en fonction/hors fonction) sert à l'activation et à la désactivation du trajet du courant. Un trajet de courant désactivé correspond à une séquence d'opérations qui écrit une valeur nulle au lieu de la valeur calculée ou à une séquence d'opérations qui laisse la valeur en mémoire inchangée. Les opérations déclenchées par l'une des opérations énumérées dépendent du relais de masquage.

Les opérations **Affectation** et **Connecteur** inscrivent la valeur 0 en mémoire si le MCR est égal à 0. Les opérations **Mettre à 1** et **Mettre à 0** ne modifient pas la valeur existante.

Opérations influencées par les zones MCR :

- # Connecteur
- = Affectation
- S Mettre à 1
- R Mettre à 0
- SR Bascule mise à 1, mise à 0
- RS Bascule mise à 0, mise à 1
- MOVE Affecter valeur

Opérations dépendant du MCR et leurs réactions à l'état de signal du MCR

Etat de signal du MCR	**Affectation, Connecteur**	**Mettre à 1, Mettre à 0**	**Affecter valeur**
0 (HORS FONCTION)	Ecrit 0. Imite un relais passant à l'état de repos en cas de perte de tension.	N'écrit rien. Imite un relais restant à l'état en vigueur en cas de perte de tension.	Ecrit 0. Imite un composant produisant une valeur de 0 en cas de perte de tension.
1 (EN FONCTION)	Exécution normale.	Exécution normale.	Exécution normale.

10.10 Remarques importantes sur l'utilisation de la fonctionnalité MCR

Attention avec les blocs dans lesquels le relais de masquage a été activé par l'instruction MCRA :

- Lorsque le relais de masquage (MCR) est hors fonction, la valeur 0 est écrite par toutes les affectation (T, =) dans les sections de programme entre relais de masquage en fonction et relais de masquage hors fonction !
- Le MCR se trouve précisément hors fonction lorsque le RLG était égal à 0 avant une instruction relais de masquage en fonction.

Danger : arrêt de l'AP ou comportement indéfini de la durée d'exécution !

Pour les calculs d'adresses, le compilateur accède également en écriture aux données locales suivant les variables temporaires définies dans VAR_TEMP. De ce fait, les séquences d'instructions suivantes mettent l'AP à l'arrêt ou conduisent à des comportements indéfinis de la durée d'exécution :

Accès à des paramètres formels

- Accès à des composants de paramètres FC complexes de type STRUCT, UDT, ARRAY, STRING
- Accès à des composants de paramètres FB complexes de type STRUCT, UDT, ARRAY, STRING de la zone IN_OUT dans un bloc admettant les multi-instances (bloc de version 2).
- Accès aux paramètres d'un FB admettant les multi-instances (bloc de version 2) lorsque leur adresse est supérieure à 8180.0.
- L'accès à un paramètre de type BLOCK_DB dans un FB admettant les multi-instances (bloc de version 2) ouvre le DB 0. Les accès ultérieurs aux données mettent la CPU à l'arrêt. Pour TIMER, COUNTER, BLOCK_FC, BLOCK_FB se sont aussi toujours T 0, Z 0, FC 0 ou FB 0 qui sont utilisés.

Transmission des paramètres

- Appels pour lesquels des paramètres sont transmis.

CONT/LOG

- Dans CONT ou LOG, les branches T et les connecteurs débutent par RLG = 0.

Remède

Séparez les instructions concernées de la dépendance par rapport au relais de masquage :

1er Désactivez le relais de masquage en utilisant l'instruction **MCRD** avant l'instruction ou le réseau concernés.

2e Activez le relais de masquage en utilisant l'instruction **MCRA** après l'instruction ou le réseau concernés.

10.11 MCR< / MCR> : Relais de masquage en fonction/hors fonction

Remarques importantes sur l'utilisation de la fonctionnalité MCR

Représentation

Relais de masquage en fonction

L'opération **Relais de masquage en fonction** (MCR<) empile le résultat logique RLG dans la pile MCR et ouvre une zone MCR. Les opérations présentées à la rubrique Fonctions du relais de masquage sont influencées à l'intérieur de cette zone par le RLG qui est sauvegardé dans la pile MCR à l'ouverture de la zone MCR.

La pile MCR, qui fonctionne selon le principe "dernier entré, premier sorti", peut contenir jusqu'à huit entrées. Si elle est pleine, l'opération **Relais de masquage en fonction** provoque une erreur de pile MCR (MCRF).

Représentation

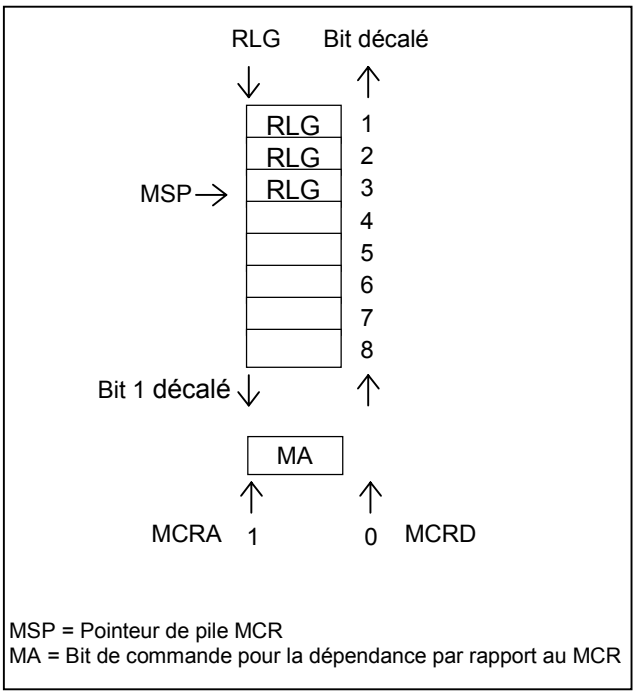
Relais de masquage hors fonction

L'opération **Relais de masquage hors fonction** (MCR>) ferme la zone MCR ouverte en dernier, en effaçant de la pile MCR l'entrée RLG qui y avait été empilée par l'opération **Relais de masquage en fonction.** L'entrée libérée à l'autre extrémité de la pile MCR, qui fonctionne selon le principe "dernier entré, premier sorti", est mise à 1.

Si la pile est déjà vide, l'opération **Relais de masquage hors fonction** provoque une erreur de pile MCR (MCRF).

Pile MCR

Le relais de masquage est commandé par une pile MCR de 1 bit de large et pouvant contenir jusqu'à huit entrées. Le relais de masquage est activé aussi longtemps que les huit entrées de la pile sont égales à 1. L'opération MCR< copie le résultat logique dans la pile MCR. L'opération MCR> extrait la dernière entrée de la pile et met l'adresse de pile libérée à 1. En cas d'erreur - lorsque plus de huit opérations MCR> se suivent ou si vous tentez d'exécuter l'opération MCR> alors que la pile MCR est vide, par exemple -, le programme affiche un message d'erreur MCRF. La surveillance de la pile MCR suit le pointeur de pile (MSP : 0 = vide, 1 = une entrée, 2 = deux entrées, ..., 8 = huit entrées).



Vous ne devez jamais utiliser l'opération **MCR<** sans l'opération **MCR>** dans votre programme. L'opération **MCR<** reprend l'état de signal du RLG pour le copier dans le bit MCR.

L'opération **MCR>** met le bit MCR à 1 inconditionnellement. Pour cette raison, toute autre opération programmée entre les opérations MCRA et MCRD fonctionne indépendamment du bit MCR.

Imbrication des opérations MCR< et MCR>

Vous pouvez imbriquer les opérations **MCR<** et **MCR>** en tenant compte de la profondeur d'imbrication maximale égale à 8. En d'autres termes, vous pouvez écrire jusqu'à huit opérations MCR< consécutives avant d'insérer une opération MCR>. Vous devez écrire le même nombre d'opérations MCR< et MCR> dans votre programme.

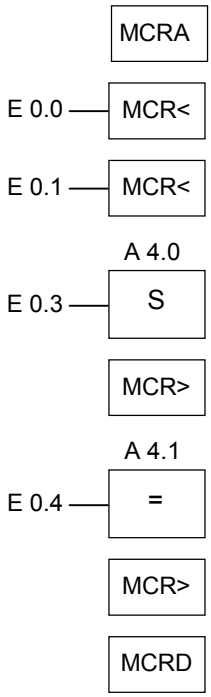
Si les opérations MCR< sont imbriquées, le programme génère le bit MCR du niveau d'imbrication inférieur. Ensuite, l'opération MCR< combine le RLG en cours au bit MCR en cours selon la table de vérité ET.

Lorsqu'une opération MCR> met fin à un niveau d'imbrication, elle extrait le bit MCR du niveau supérieur.

Mot d'état

	RB	**BI1**	**BI0**	**DEB**	**DM**	**OU**	**ETAT**	**RLG**	**/PI**
Ecriture	-	-	-	-	-	0	1	-	0

Exemple



Lorsque l'opération MCRA active la fonction MCR, vous pouvez créer jusqu'à huit zones MCR imbriquées. Dans notre exemple, il y en a deux. La première opération MCR> va de pair avec la seconde opération MCR<. Toutes les opérations entre le deuxième jeu MCR (MCR<MCR>) appartiennent à la seconde zone MCR. Les opérations sont exécutées comme suit :

- Si E 0.0 = 1, l'état de signal de E 0.4 est affecté à la sortie A 4.1.
- Si E 0.0 = 0, la sortie A 4.1 est mise à 0 quel que soit l'état de signal à l'entrée E 0.4. La sortie A 4.0 reste inchangée quel que soit l'état de signal à l'entrée E 0.3.
- Si E 0.0 et E 0.1 = 1, la sortie A 4.0 est mise à 1 si l'état de signal est 1 à l'entrée E 0.3 et l'état de signal de E 0.4 est affecté à A 4.1.
- Si E 0.1 = 0, la sortie A 4.0 reste inchangée quel que soit l'état de signal aux entrées E 0.3 et E 0.0.

10.12 MCRA / MCRD : Activer/désactiver relais de masquage

Remarques importantes sur l'utilisation de la fonctionnalité MCR

Représentation

Activer relais de masquage

L'opération **Activer relais de masquage** active la relation de dépendance entre le relais de masquage (MCR, Master Control Relay) et les commandes suivantes. Vous pouvez, après cette commande, faire appel aux opérations **Relais de masquage en fonction et Relais de masquage hors fonction** pour programmer des zones MCR (voir la rubrique Relais de masquage en fonction/hors fonction). Lorsque votre programme active une zone MCR, toutes les actions MCR dépendent du contenu de la pile MCR.

Représentation

Désactiver relais de masquage

L'opération **Désactiver relais de masquage** désactive la relation de dépendance entre le relais de masquage et les commandes suivantes. Après cette opération, vous ne pouvez plus programmer de zones MCR. Lorsque votre programme désactive une zone MCR, le courant circule toujours dans le relais de masquage, indépendamment des entrées figurant dans la pile MCR.

La pile MCR et le bit MA qui commande la relation de dépendance se réfèrent à un niveau de séquence précis ; vous devez donc les sauvegarder et les restaurer à chaque changement de niveau. Au début de chaque niveau de séquence, les bits 1 à 8 de saisie MCR sont initialisés à 1, le pointeur de pile MCR et le bit MA sont initialisés à 0.

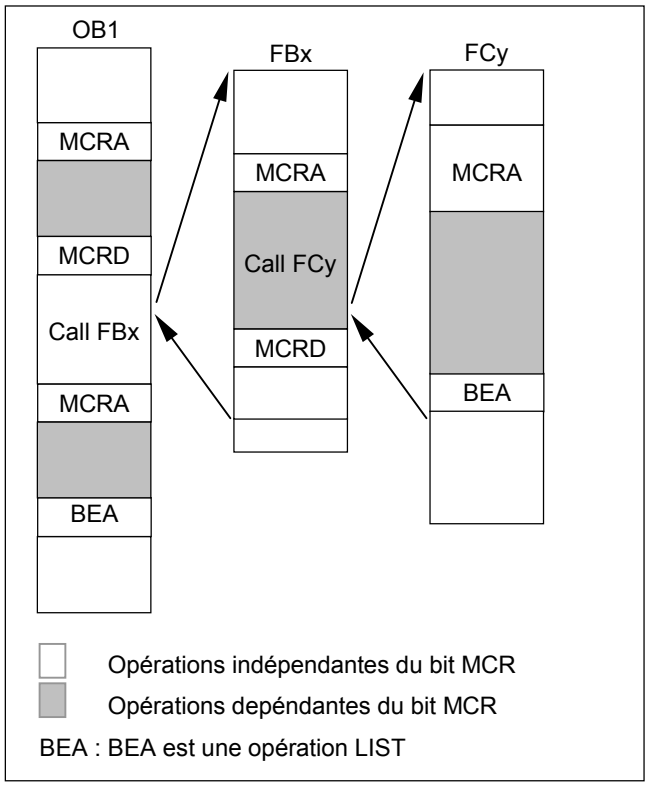
La pile MCR est transmise de bloc en bloc. Le bit MA est sauvegardé à chaque appel de bloc et mis à 0. Il est restauré à la fin du bloc.

Le relais de masquage peut être réalisé de façon à optimiser les temps d'exécution nécessaires aux CPU génératrices de code. L'optimisation s'explique par le fait que la dépendance par rapport au relais de masquage n'est pas transmise au bloc, mais qu'elle doit être activée explicitement par une opération MCRA. Une CPU génératrice de code reconnaît l'opération MCRA et génère le code supplémentaire nécessaire à l'évaluation de la pile MCR jusqu'à ce qu'elle reconnaisse une opération MCRD ou la fin du bloc. Le temps d'exécution n'est donc pas allongé pour les opérations programmées hors de la zone délimitée par MCRA et MCRD.

Vous ne devez jamais utiliser l'opération **MCRA** sans l'opération **MCRD** dans votre programme.

Activation et désactivation d'une zone MCR

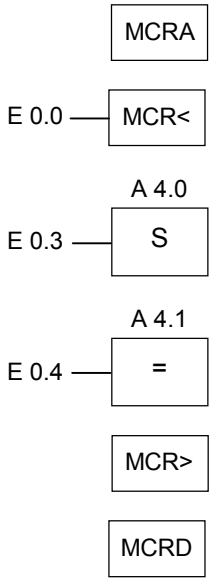
Les opérations programmées entre MCRA et MCRD dépendent de l'état de signal du bit MCR, contrairement à celles programmées en dehors de cette séquence. S'il manque une opération MCRD, les opérations programmées entre MCRA et BEA dépendent du bit MCR.



Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	-	-	-	-	-	-	-	-

Exemple



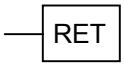
L'opération MCRA active la fonction MCR jusqu'au prochain MCRD. Les opérations entre MCR< et MCR> sont traitées en fonction du bit MA (E 0.0 ici) :

- Si l'état de signal de l'entrée E 0.0 est 1 :
  - la sortie A 4.0 est mise à 1 si l'état de signal est 1 à l'entrée E 0.3 ;
  - la sortie A 4.0 reste inchangée si l'état de signal est 0 à l'entrée E 0.3 ;
  - l'état de signal à l'entrée E 0.4 est affecté à la sortie A 4.1.
- Si l'état de signal de l'entrée E 0.0 est 0 :
  - la sortie A 4.0 reste inchangée quel que soit l'état de signal à l'entrée E 0.3 ;
  - la sortie A 4.1 est à 0 quel que soit l'état de signal à l'entrée E 0.4.

Vous devez programmer dans les blocs mêmes la relation de dépendance au MCR des fonctions et des blocs fonctionnels. En effet, si vous appelez une fonction ou un bloc fonctionnel à partir d'une séquence MCRA-MCRD, toutes les instructions à l'intérieur de cette séquence ne dépendent pas automatiquement du bit MCR. Pour que les instructions contenues dans le bloc appelé dépendent du MCR, il faut utiliser l'opération MCRA dans le bloc appelé.

10.13 RET : Retour

Représentation



Description

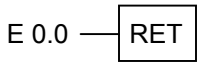
L'opération **Retour** permet de quitter des blocs conditionnellement.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	*	-	-	-	0	0	1	1	0

* L'opération étant mappée de manière interne sur la séquence "SAVE; BEB;", le bit RB est influencé lui aussi.

Exemple



Le bloc est abandonné si l'état de signal est 1 à l'entrée E 0.0.

11 Opérations de décalage et de rotation

11.1 Opérations de décalage

11.1.1 Vue d'ensemble des opérations de décalage

Description

Les opérations de décalage permettent de décaler bit par bit le contenu de l'entrée IN vers la gauche ou vers la droite (voir Registres de la CPU). Un décalage de n bits vers la gauche multiplie le contenu de l'entrée IN par 2n ; un décalage de n bits vers la droite le divise par 2n. Si, par exemple, vous décalez de 3 bits vers la gauche l'équivalent binaire de la valeur décimale 3, vous obtenez l'équivalent binaire de la valeur décimale 24. Si vous décalez de 2 bits vers la droite l'équivalent binaire de la valeur décimale 16, vous obtenez l'équivalent binaire de la valeur décimale 4.

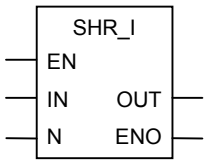
Le nombre de bits de décalage est précisé dans le paramètre d'entrée N. Les positions binaires libérées par l'opération de décalage sont soit complétées par des zéros, soit par l'état de signal du bit de signe (0 signifie positif et 1 négatif). L'état de signal du bit décalé en dernier est chargé dans le bit BI1 du mot d'état. Les bits BI0 et DEB du mot d'état sont remis à 0. Vous pouvez évaluer le bit BI1 à l'aide d'opérations de saut.

Vous disposez des opérations de décalage suivantes :

- SHR_I : Décalage vers la droite d'un entier de 16 bits
- SHR_DI : Décalage vers la droite d'un entier de 32 bits
- SHL_W : Décalage vers la gauche d'un mot
- SHR_W : Décalage vers la droite d'un mot
- SHL_DW : Décalage vers la gauche d'un double mot
- SHR_DW : Décalage vers la droite d'un double mot

11.1.2 SHR_I : Décalage vers la droite d'un entier de 16 bits

Représentation

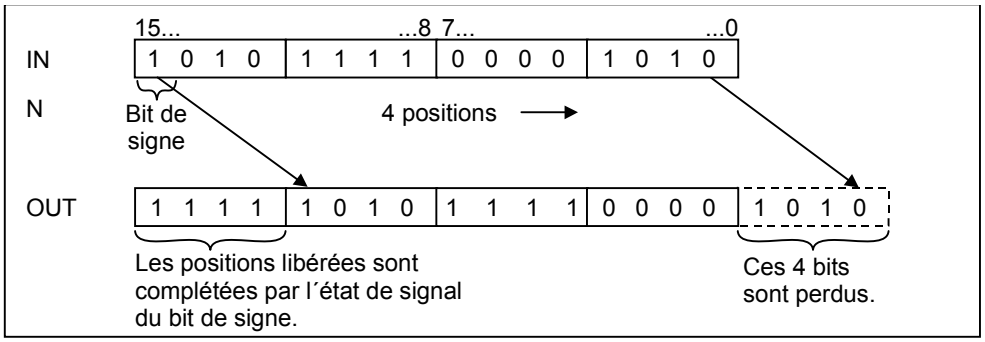


Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, L, D, T, Z	Entrée de validation
IN	INT	E, A, M, L, D	Valeur à décaler
N	WORD	E, A, M, L, D	Nombre de bits de décalage
OUT	INT	E, A, M, L, D	Résultat du décalage
ENO	BOOL	E, A, M, L, D	Sortie de validation

Description

L'opération **Décalage vers la droite d'un entier de 16 bits** est activée si l'état de signal est 1 à l'entrée de validation EN. Elle décale bit par bit vers la droite les bits 0 à 15 de l'entrée IN. Le nombre de bits de décalage est indiqué dans l'entrée N. Si N est supérieur à 16, la commande agit comme si N était égal à 16. Les positions binaires libérées à gauche sont complétées selon l'état de signal du bit 15 (bit de signe du nombre entier de 16 bits) : par des zéros si le nombre est positif et par des uns s'il est négatif. Le résultat du décalage est rangé dans la sortie OUT.

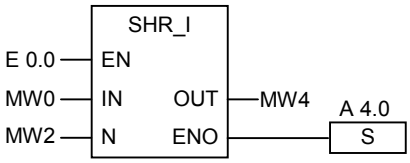
Lorsque N est différent de zéro, l'opération met les bits BI0 et DEB du mot d'état à 0. ENO possède le même état de signal que EN.



Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	X	X	X	X	-	X	X	X	1

Exemple



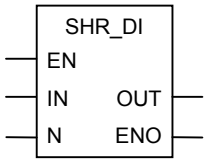
L'opération est activée si l'état de signal est 1 à l'entrée E 0.0.

Le mot de mémento MW0 est décalé vers la droite du nombre de bits précisé dans MW2.

Le résultat est rangé dans MW4. La sortie A 4.0 est mise à 1.

11.1.3 SHR_DI : Décalage vers la droite d'un entier de 32 bits

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, L, D, T, Z	Entrée de validation
IN	DINT	E, A, M, L, D	Valeur à décaler
N	WORD	E, A, M, L, D	Nombre de bits de décalage
OUT	DINT	E, A, M, L, D	Résultat du décalage
ENO	BOOL	E, A, M, L, D	Sortie de validation

Description

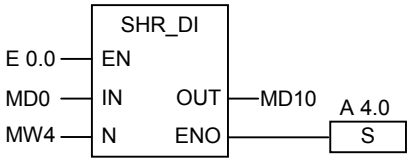
L'opération **Décalage vers la droite d'un entier de 32 bits** est activée si l'état de signal est 1 à l'entrée de validation EN. Elle décale bit par bit vers la droite les bits 0 à 31 de l'entrée IN. Le nombre de bits de décalage est indiqué dans l'entrée N. Si N est supérieur à 32, la commande agit comme si N était égal à 32. Les positions binaires libérées à gauche sont complétées selon l'état de signal du bit 31 (bit de signe du nombre entier de 32 bits) : par des zéros si le nombre est positif et par des uns s'il est négatif. Le résultat du décalage est rangé dans la sortie OUT.

Lorsque N est différent de zéro, l'opération met les bits BI0 et DEB du mot d'état à 0. ENO possède le même état de signal que EN.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	X	X	X	X	-	X	X	X	1

Exemple



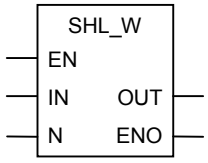
L'opération est activée si l'état de signal est 1 à l'entrée E 0.0.

Le double mot de mémento MD0 est décalé vers la droite du nombre de bits précisé dans MW4.

Le résultat est rangé dans MD10. La sortie A 4.0 est mise à 1.

11.1.4 SHL_W : Décalage vers la gauche d'un mot

Représentation



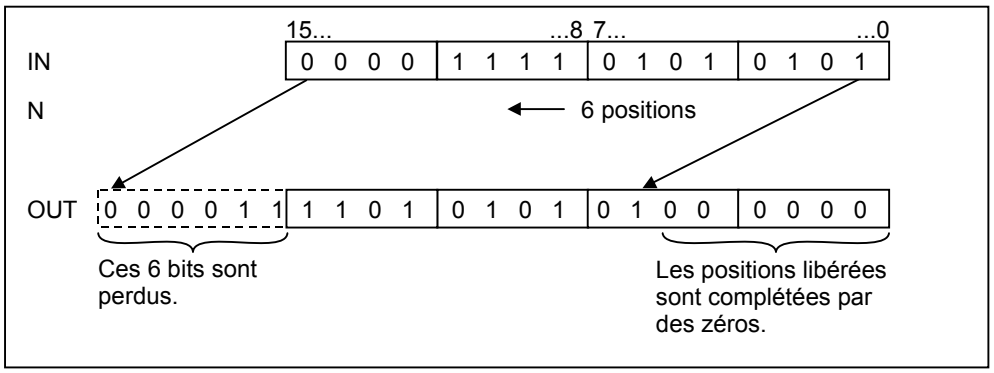
Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, L, D, T, Z	Entrée de validation
IN	WORD	E, A, M, L, D	Valeur à décaler
N	WORD	E, A, M, L, D	Nombre de bits de décalage
OUT	WORD	E, A, M, L, D	Résultat du décalage
ENO	BOOL	E, A, M, L, D	Sortie de validation

Description

L'opération **Décalage vers la gauche d'un mot** est activée si l'état de signal est 1 à l'entrée de validation EN. Elle décale bit par bit vers la gauche les bits 0 à 15 de l'entrée IN.

Le nombre de bits de décalage est indiqué dans l'entrée N. Si N est supérieur à 16, la commande inscrit 0 dans la sortie OUT et met à 0 les bits BI0 et DEB du mot d'état. Les positions binaires libérées à droite sont complétées par des zéros. Le résultat du décalage est rangé dans la sortie OUT.

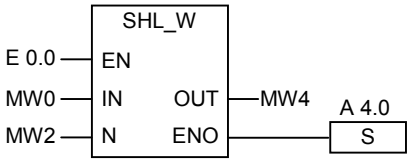
Lorsque N est différent de zéro, l'opération met les bits BI0 et DEB du mot d'état à 0. ENO possède le même état de signal que EN.



Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	X	X	X	X	-	X	X	X	1

Exemple



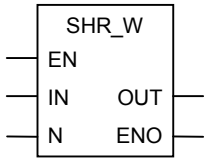
L'opération est activée si l'état de signal est 1 à l'entrée E 0.0.

Le mot de mémento MW0 est décalé vers la gauche du nombre de bits précisé dans MW2.

Le résultat est rangé dans MW4. La sortie A 4.0 est mise à 1.

11.1.5 SHR_W : Décalage vers la droite d'un mot

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, L, D, T, Z	Entrée de validation
IN	WORD	E, A, M, L, D	Valeur à décaler
N	WORD	E, A, M, L, D	Nombre de bits de décalage
OUT	WORD	E, A, M, L, D	Résultat du décalage
ENO	BOOL	E, A, M, L, D2	Sortie de validation

Description

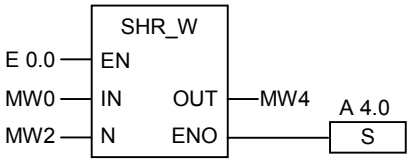
L'opération **Décalage vers la droite d'un mot** est activée si l'état de signal est 1 à l'entrée de validation EN. Elle décale bit par bit vers la droite les bits 0 à 15 de l'entrée IN. Le nombre de bits de décalage est indiqué dans l'entrée N. Si N est supérieur à 16, la commande inscrit 0 dans la sortie OUT et met à 0 les bits BI0 et DEB du mot d'état. Les positions binaires libérées à gauche sont complétées par des zéros. Le résultat du décalage est rangé dans la sortie OUT.

Lorsque N est différent de zéro, l'opération met toujours les bits BI0 et DEB du mot d'état à 0. ENO possède le même état de signal que EN.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	X	X	X	X	-	X	X	X	1

Exemple



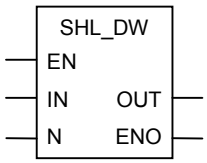
L'opération est activée si l'état de signal est 1 à l'entrée E 0.0.

Le mot de mémento MW0 est décalé vers la droite du nombre de bits précisé dans MW2.

Le résultat est rangé dans MW4. La sortie A 4.0 est mise à 1.

11.1.6 SHL_DW : Décalage vers la gauche d'un double mot

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, L, D, T, Z	Entrée de validation
IN	DWORD	E, A, M, L, D	Valeur à décaler
N	WORD	E, A, M, L, D	Nombre de bits de décalage
OUT	DWORD	E, A, M, L, D	Résultat du décalage
ENO	BOOL	E, A, M, L, D	Sortie de validation

Description

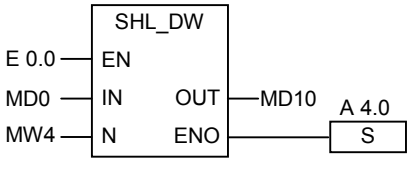
L'opération **Décalage vers la gauche d'un double mot** est activée si l'état de signal est 1 à l'entrée de validation EN. Elle décale bit par bit vers la gauche les bits 0 à 31 de l'entrée IN. Le nombre de bits de décalage est indiqué dans l'entrée N. Si N est supérieur à 32, la commande inscrit 0 dans la sortie OUT et met à 0 les bits BI0 et DEB du mot d'état. Les positions binaires libérées à droite sont complétées par des zéros. Le résultat du décalage est rangé dans la sortie OUT.

Lorsque N est différent de zéro, l'opération met les bits BI0 et DEB du mot d'état à 0. ENO possède le même état de signal que EN.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	X	X	X	X	-	X	X	X	1

Exemple



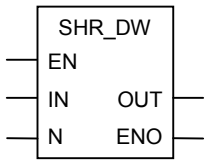
L'opération est activée si l'état de signal est 1 à l'entrée E 0.0.

Le double mot de mémento MD0 est décalé vers la gauche du nombre de bits précisé dans MW4.

Le résultat est rangé dans MD10. La sortie A 4.0 est mise à 1.

11.1.7 SHR_DW : Décalage vers la droite d'un double mot

Représentation

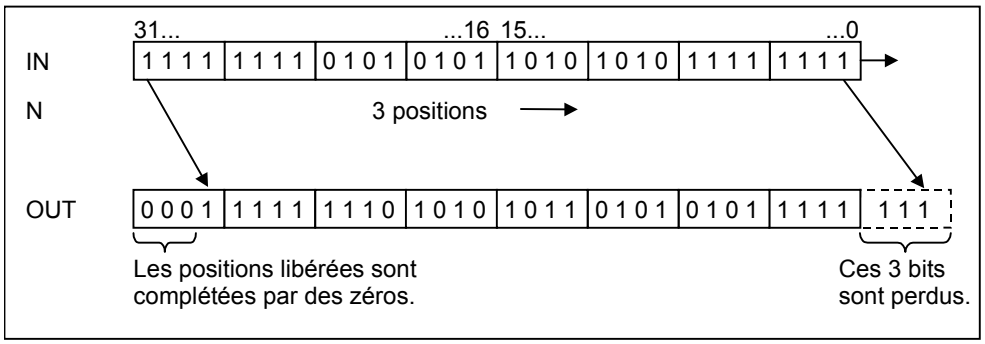


Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, L, D, T, Z	Entrée de validation
IN	DWORD	E, A, M, L, D	Valeur à décaler
N	WORD	E, A, M, L, D	Nombre de bits de décalage
OUT	DWORD	E, A, M, L, D	Résultat du décalage
ENO	BOOL	E, A, M, L, D	Sortie de validation

Description

L'opération **Décalage vers la droite d'un double mot** est activée si l'état de signal est 1 à l'entrée de validation EN. Elle décale bit par bit vers la droite les bits 0 à 31 de l'entrée IN. Le nombre de bits de décalage est indiqué dans l'entrée N. Si N est supérieur à 32, la commande inscrit 0 dans la sortie OUT et met à 0 les bits BI0 et DEB du mot d'état. Les positions binaires libérées à gauche sont complétées par des zéros. Le résultat du décalage est rangé dans la sortie OUT.

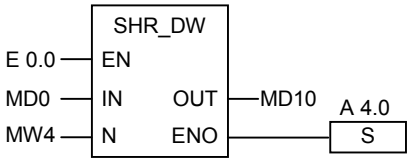
Lorsque N est différent de zéro, l'opération met les bits BI0 et DEB du mot d'état à 0. ENO possède le même état de signal que EN.



Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	X	X	X	X	-	X	X	X	1

Exemple



L'opération est activée si l'état de signal est 1 à l'entrée E 0.0.

Le double mot de mémento MD0 est décalé vers la droite du nombre de bits précisé dans MW4.

Le résultat est rangé dans MD10. La sortie A 4.0 est mise à 1.

11.2 Opérations de rotation

11.2.1 Vue d'ensemble des opérations de rotation

Description

Les opérations de rotation permettent d'effectuer la rotation bit par bit vers la droite ou vers la gauche du contenu entier de l'entrée IN. Les positions binaires libérées sont complétées par l'état de signal des bits qui ont été décalés hors de l'entrée IN.

Le nombre de bits de rotation est précisé dans le paramètre d'entrée N.

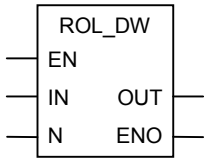
Selon l'opération choisie, la rotation s'effectue via le bit BI1 du mot d'état. Le bit BI0 du mot d'état est remis à 0.

Vous disposez des opérations de rotation suivantes :

- ROL_DW : Rotation vers la gauche d'un double mot
- ROR_DW : Rotation vers la droite d'un double mot

11.2.2 ROL_DW : Rotation vers la gauche d'un double mot

Représentation

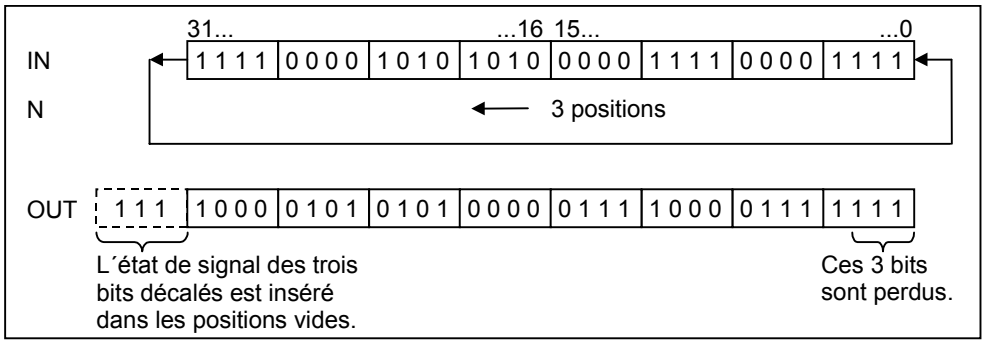


Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, L, D, T, Z	Entrée de validation
IN	DWORD	E, A, M, L, D	Valeur objet de la rotation
N	WORD	E, A, M, L, D	Nombre de bits de rotation
OUT	DWORD	E, A, M, L, D	Résultat de la rotation
ENO	BOOL	E, A, M, L, D	Sortie de validation

Description

L'opération **Rotation vers la gauche d'un double mot** est activée si l'état de signal est 1 à l'entrée de validation EN. Elle déclenche la rotation bit par bit vers la gauche du contenu entier de l'entrée IN. Le nombre de bits de rotation est indiqué dans l'entrée N. Si N est supérieur à 32, le double mot effectue une rotation de ((N-1) modulo 32) + 1 positions vers la gauche. Les positions binaires libérées à droite sont complétées par l'état de signal des bits ayant effectué la rotation. Le résultat de la rotation est rangé dans la sortie OUT.

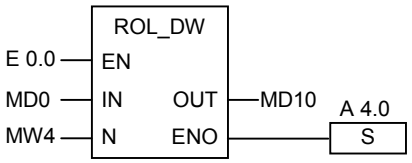
Lorsque N est différent de zéro, l'opération met les bits BI0 et DEB du mot d'état à 0. ENO possède le même état de signal que EN.



Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	X	X	X	X	-	X	X	X	1

Exemple



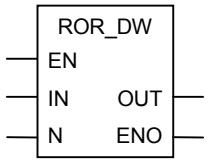
L'opération est activée si l'état de signal est 1 à l'entrée E 0.0.

Le double mot de mémento MD0 fait l'objet d'une rotation vers la gauche du nombre de bits précisé dans MW4.

Le résultat est rangé dans MD10. La sortie A 4.0 est mise à 1.

11.2.3 ROR_DW : Rotation vers la droite d'un double mot

Représentation

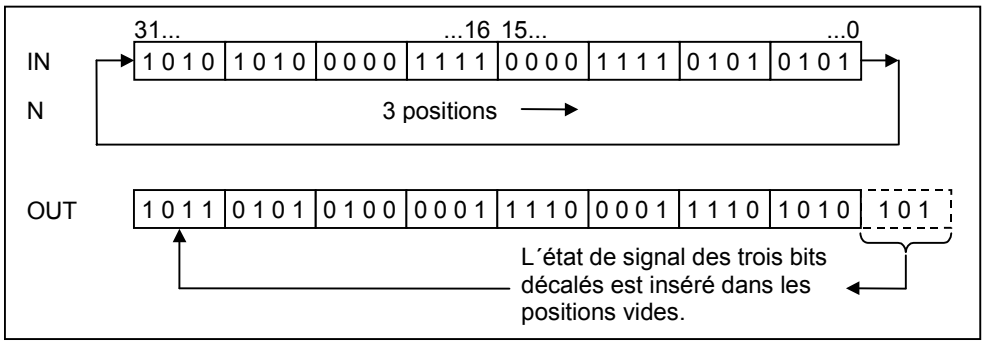


Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, L, D, T, Z	Entrée de validation
IN	DWORD	E, A, M, L, D	Valeur objet de la rotation
N	WORD	E, A, M, L, D	Nombre de bits de rotation
OUT	DWORD	E, A, M, L, D	Résultat de la rotation
ENO	BOOL	E, A, M, L, D	Sortie de validation

Description

L'opération **Rotation vers la droite d'un double mot** est activée si l'état de signal est 1 à l'entrée de validation EN. Elle déclenche la rotation bit par bit vers la droite du contenu entier de l'entrée IN. Le nombre de bits de rotation est indiqué dans l'entrée N dont la valeur peut être comprise entre 0 et 31. Si N est supérieur à 32, le double mot effectue une rotation de ((N-1) modulo 32) + 1 positions vers la droite. Les positions binaires libérées à gauche sont complétées par l'état de signal des bits ayant effectué la rotation. Le résultat de la rotation est rangé dans la sortie OUT.

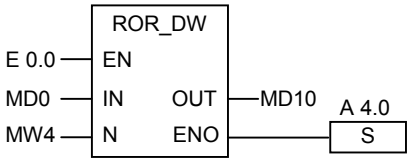
Lorsque N est différent de zéro, l'opération met les bits BI0 et DEB du mot d'état à 0. ENO possède le même état de signal que EN.



Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	X	X	X	X	-	X	X	X	1

Exemple



L'opération est activée si l'état de signal est 1 à l'entrée E 0.0.

Le double mot de mémento MD0 fait l'objet d'une rotation vers la droite du nombre de bits précisé dans MW4.

Le résultat est rangé dans MD10. La sortie A 4.0 est mise à 1.

12 Opérations sur bits d'état

12.1 Vue d'ensemble des opérations sur bits d'état

Description

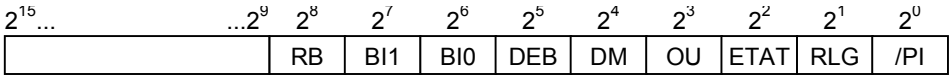
Les opérations sur bits d'état sont des opérations combinatoires sur bits qui utilisent les bits du mot d'état. Ces opérations réagissent à l'une des conditions suivantes indiquées par un ou plusieurs bits du mot d'état :

- Le bit de résultat binaire RB est mis à 1 (son état de signal est égal à 1).
- Le résultat d'une opération arithmétique est : == 0, <> 0, > 0, < 0, >= 0, <= 0.
- Le résultat d'une opération arithmétique est illicite (UO).
- Un débordement (OV) s'est produit lors d'une opération arithmétique ou un débordement mémorisé (OS).

Dans une opération ET, les opérations sur bits d'état combinent le résultat de leur interrogation d'état de signal au résultat logique précédent selon la table de vérité ET ; dans une opération OU, elles le combinent selon la table de vérité OU.

Mot d'état

Le mot d'état est un registre dans la mémoire de votre CPU contenant des bits auxquels vous pouvez accéder dans les opérandes de combinaisons sur bits et sur mots. La figure suivante présente la structure du mot d'état.

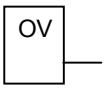


Vous pouvez évaluer les bits du mot d´état :

- dans les opérations sur nombres entiers,
- dans les opérations sur nombres à virgule flottante.

12.2 OV : Bit d'anomalie "Débordement"

Représentation



Description

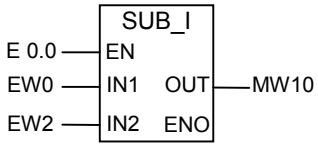
L'opération **Bit d'anomalie Débordement** permet de détecter un débordement dans une opération arithmétique. Si, après une opération arithmétique, le résultat se situe hors de la plage négative ou positive autorisée, le bit de débordement (DEB) du mot d'état est mis à 1. L'opération **Bit d'anomalie Débordement** interroge l'état de signal de ce bit, qui est remis à 0 par des opérations arithmétiques exécutées sans erreur.

Mot d'état

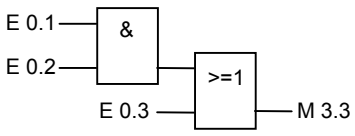
	RB	**BI1**	**BI0**	**DEB**	**DM**	**OU**	**ETAT**	**RLG**	**/PI**
Ecriture	-	-	-	-	-	X	X	X	1

Exemple

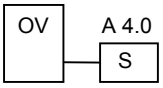
Réseau 1



Réseau 2



Réseau 3



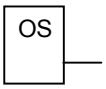
La boîte SUB_I est activée si l'état de signal est 1 à l'entrée E 0.0. Si le résultat de l'opération arithmétique EW0 - EW2 est hors de la plage autorisée pour un nombre entier de 16 bits, le bit DEB du mot d'état est mis à 1.

Le résultat d'une interrogation d'état de signal pour DEB est égal à 1. La sortie A 4.0 est mise à 1 si l'interrogation de DEB donne 1 et que le RLG du deuxième réseau soit égal à 1 (c'est-à-dire si le RLG avant la sortie A 4.0 est 1).

Si l'état de signal est 0 à l'entrée E 0.0 (désactivée), EN et ENO ont tous deux l'état de signal 0. Si EN est à 1 (activé) et que le résultat de l'opération arithmétique se situe hors de la plage autorisée, ENO est mis à 0.

12.3 OS : Bit d'anomalie "Débordement mémorisé"

Représentation



Description

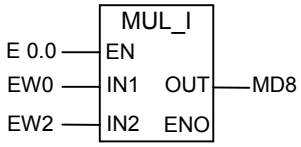
L'opération **Bit d'anomalie Débordement mémorisé** permet de détecter un débordement mémorisé (DM) dans une opération arithmétique. Si, après une opération arithmétique, le résultat se situe hors de la plage négative ou positive autorisée, le bit DM du mot d'état est mis à 1. L'opération **Bit d'anomalie Débordement mémorisé** interroge l'état de signal de ce bit. Contrairement à ce qui se passe pour le bit DEB (débordement), le bit DM reste à 1 lorsque des opérations arithmétiques sont exécutées sans erreur (voir la rubrique Bit d'anomalie "Débordement").

Mot d'état

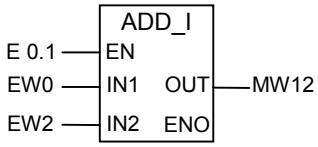
	RB	**BI1**	**BI0**	**DEB**	**DM**	**OU**	**ETAT**	**RLG**	**/PI**
Ecriture	-	-	-	-	-	X	X	X	1

Exemple

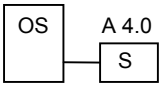
Réseau 1



Réseau 2



Réseau 3



La boîte MUL_I est activée si l'état de signal est 1 à l'entrée E 0.0. La boîte ADD_I est activée si l'état de signal est 1 à l'entrée E 0.1. Si le résultat de l'une de ces opérations arithmétiques est hors de la plage autorisée pour un nombre entier de 16 bits, le bit DM du mot d'état est mis à 1.

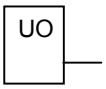
Le résultat d'une interrogation d'état de signal pour DM est égal à 1 et la sortie A 4.0 est mise à 1.

Réseau 1 : Si l'état de signal est 0 à l'entrée E 0.0 (désactivée), EN et ENO ont tous deux l'état de signal 0. Si EN est à 1 (activé) et que le résultat de l'opération arithmétique se situe hors de la plage autorisée, ENO est mis à 0.

Réseau 2 : Si l'état de signal est 0 à l'entrée E 0.1 (désactivée), EN et ENO ont tous deux l'état de signal 0. Si EN est à 1 (activé) et que le résultat de l'opération arithmétique se situe hors de la plage autorisée, ENO est mis à 0.

12.4 UO : Bit d'anomalie "Opération illicite"

Représentation



Description

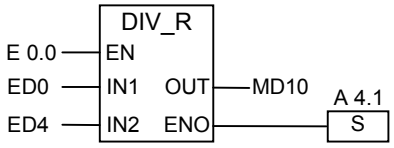
L'opération **Bit d'anomalie Opération illicite** permet de déterminer si le résultat d'une opération arithmétique sur nombres réels est ou non illicite, c'est-à-dire si l'une des valeurs de l'opération arithmétique n'est pas un nombre à virgule flottante autorisé. Les bits indicateurs BI1 et BI0 du mot d'état sont évalués à cet effet. Si le résultat d'une opération arithmétique est illicite (UO), l'interrogation de l'état de signal fournit un résultat égal à 1. Si la combinaison dans BI1 et BI0 ne signifie pas illicite, le résultat de l'interrogation de l'état de signal est 0.

Mot d'état

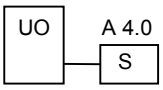
	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	-	-	-	-	X	X	X	1

Exemple

Réseau 1



Réseau 2



La boîte DIV_R est activée si l'état de signal est 1 à l'entrée E 0.0. Si la valeur de ED0 ou de ED4 n'est pas un nombre à virgule flottante correct, l'opération arithmétique est illicite. Si EN est à 1 (activé) et qu'une erreur se produise lors de l'exécution de la fonction DIV_R, ENO est mis à 0.

La sortie A 4.0 est mise à 1 si l'opération DIV_R est exécutée mais que l'une des valeurs dans l'opération arithmétique n'est pas un nombre à virgule flottante correct. Si l'état de signal est 0 à l'entrée E 0.0 (désactivée), EN et ENO ont tous deux l'état de signal 0.

12.5 BIE : Bit d'anomalie "Registre RB"

Représentation

Anglaise **Allemande**

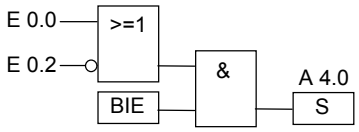
Description

L'opération **Bit d'anomalie Registre RB** permet d'interroger l'état du bit de résultat binaire RB du mot d'état (voir aussi Registres de la CPU).

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	-	-	-	-	X	X	X	1

Exemple



La sortie A 4.0 est mise à 1 si l'état de signal est 1 à l'entrée E 0.0 **ou** à 0 à l'entrée E 0.2 et - en plus de ce RLG - si l'état de signal du bit RB est 1.

12.6 <> 0 : Bits de résultat

Représentation

== 0 —	L'opération **Bit de résultat pour égal à 0** détermine si le résultat d'une opération arithmétique est ou non égal à 0.
<> 0 —	L'opération **Bit de résultat pour différent de 0** détermine si le résultat d'une opération arithmétique est ou non différent de 0.
> 0 —	L'opération **Bit de résultat pour supérieur à 0** détermine si le résultat d'une opération arithmétique est ou non supérieur à 0.
< 0 —	L'opération **Bit de résultat pour inférieur à 0** détermine si le résultat d'une opération arithmétique est ou non inférieur à 0.
>= 0 —	L'opération **Bit de résultat pour supérieur ou égal à 0** détermine si le résultat d'une opération arithmétique est ou non supérieur ou égal à 0.
<= 0 —	L'opération **Bit de résultat pour inférieur ou égal à 0** détermine si le résultat d'une opération arithmétique est ou non inférieur ou égal à 0.

Description

Les opérations **Bits de résultat** permettent de déterminer la relation par rapport à 0 (== 0, <> 0, > 0, < 0, >= 0, <= 0) du résultat d'une opération arithmétique. Les bits indicateurs BI1 et BI0 du mot d'état sont évalués à cet effet. Si la condition de comparaison précisée dans l'opérande est satisfaite, le résultat de l'interrogation d'état de signal est 1.

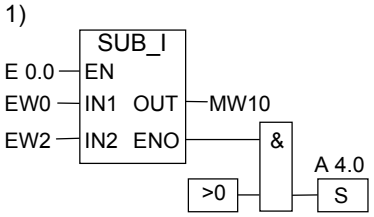
Dans une opération ET, le résultat de l'interrogation est combiné au résultat logique précédent selon la table de vérité ET ; dans une opération OU, il est combiné au résultat logique précédent selon la table de vérité OU.

Mot d'état

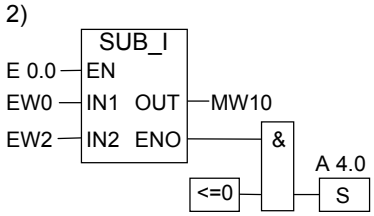
	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	-	-	-	-	X	X	X	1

Exemple

1)



2)



La boîte SUB_I est activée si l'état de signal est 1 à l'entrée E 0.0. Si la valeur de EW0 est supérieure à celle de EW2, le résultat de l'opération arithmétique EW0 - EW2 est supérieur à 0.
Si EN est à 1 et qu'une erreur se produise lors de l'exécution de la fonction SUB_I, ENO est mis à 0.

1) La sortie A 4.0 est mise à 1 si la fonction s'exécute sans erreur et si le résultat est supérieur à 0. Si l'état de signal est 0 à l'entrée E 0.0 (désactivée), EN et ENO ont tous deux l'état de signal 0.

2) La sortie A 4.0 est mise à 1 si la fonction s'exécute sans erreur et si le résultat est inférieur ou égal à 0. Si l'état de signal est 0 à l'entrée E 0.0 (désactivée), EN et ENO ont tous deux l'état de signal 0.

13 Opérations de temporisation

13.1 Vue d'ensemble des opérations de temporisation

Description

Vous trouverez des informations sur le réglage et la sélection de la bonne temporisation sous "Adresse d´une temporisation en mémoire et composants d´une temporisation".

Vous disposez des opérations de temporisations suivantes :

- S_IMPULS : Paramétrer et démarrer une temporisation sous forme d'impulsion
- S_VIMP : Paramétrer et démarrer une temporisation sous forme d'impulsion prolongée
- S_EVERZ : Paramétrer et démarrer une temporisation sous forme de retard à la montée
- S_SEVERZ : Paramétrer et démarrer une temporisation sous forme de retard à la montée mémorisé
- S_AVERZ : Paramétrer et démarrer une temporisation sous forme de retard à la retombée

- SI : Temporisation sous forme d'impulsion
- SV : Temporisation sous forme d'impulsion prolongée
- SE : Temporisation sous forme de retard à la montée
- SS : Temporisation sous forme de retard à la montée mémorisé
- SA : Temporisation sous forme de retard à la retombée

13.2 Adresse d'une temporisation en mémoire et composants d'une temporisation

Zone de mémoire

Les temporisations disposent d'une zone de mémoire réservée dans votre CPU, avec un mot de 16 bits pour chaque opérande de temporisation. La programmation en LOG permet d'utiliser jusqu'à 256 temporisations. Le nombre de mots de temporisations disponibles dans votre CPU est mentionné dans les caractéristiques techniques de cette dernière.

Les fonctions suivantes accèdent à la zone de mémoire des temporisations :

- opérations de temporisation,
- mise à jour des mots de temporisation via un générateur d'impulsions.
  Cette fonction décrémente, à l'état de fonctionnement "Marche" (RUN) de la CPU, une valeur donnée d'une unité dans un intervalle défini par la base de temps, et ce jusqu'à ce que la valeur de temps égale zéro.

Valeur de temps

La valeur de temps se trouve, en code binaire, dans les bits 0 à 9 du mot de temporisation ; elle indique un nombre d'unités. L'actualisation de la temporisation décrémente la valeur de temps d'une unité dans un intervalle défini par la base de temps, et ce jusqu'à ce que la valeur de temps égale zéro.

Vous pouvez charger une valeur de temps prédéfinie en utilisant l'un des deux formats suivants :

- S5T#a**H**_b**M**_c**S**_d**MS**
  - H (heures), M (minutes), S (secondes) et MS (millisecondes) ;
    a, b, c, d sont definies par l'utilisateur
  - la base de temps est choisie automatiquement et la valeur est arrondie au nombre inférieur le plus proche avec cette base de temps.

La valeur de temps maximale que vous pouvez indiquer est égale à 9 990 secondes ou 2H_46M_30S.
Exemples :

S5TIME#4S = 4 secondes
s5t#2h_15m = 2 heures et 15 minutes
S5T#1H_12M_18S = 1 heure, 12 minutes et 18 secondes

Base de temps

Les bits 12 et 13 du mot de temporisation contiennent la base de temps en code binaire. La base de temps détermine à quel intervalle la valeur de temps sera décrémentée. La base de temps minimale est égale à 10 ms ; la base de temps maximale à 10 s.

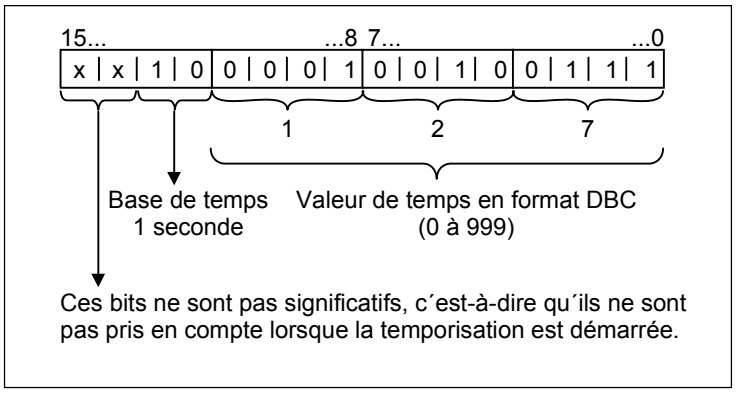
Base de temps	Code binaire de la base de temps
10 ms	00
100 ms	01
1 s	10
10 s	11

Comme les valeurs de temps sont mémorisées dans un seul intervalle de temps, celles qui ne sont pas un multiple exact de cet intervalle sont tronquées. Les valeurs dont la résolution est trop grande pour la plage considérée sont arrondies de façon à obtenir la plage voulue, mais pas avec avec la résolution souhaitée. Le tableau suivant présente les résolutions possibles avec les plages correspondantes.

Résolution	Plage
0,01 seconde	10MS à 9S_990MS
0,1 seconde	100MS à 1M_39S_900MS
1 seconde	1S à 16M_39S
10 secondes	10S à 2HR_46M_30S

Configuration des bits dans la cellule de temporisation

Lorsqu'une temporisation est démarrée, le contenu de la cellule de temporisation est utilisé comme valeur de temps. Les bits 0 à 11 de la cellule de temporisation contiennent la valeur de temps en format décimal codé binaire (format DCB : chaque groupe de quatre bits contient le code binaire d'une valeur décimale). Les bits 12 et 13 contiennent la base de temps en code binaire. La figure suivante montre le contenu de la cellule de temporisation dans laquelle vous avez chargé la valeur de temps 127 avec une base de temps de 1 seconde.

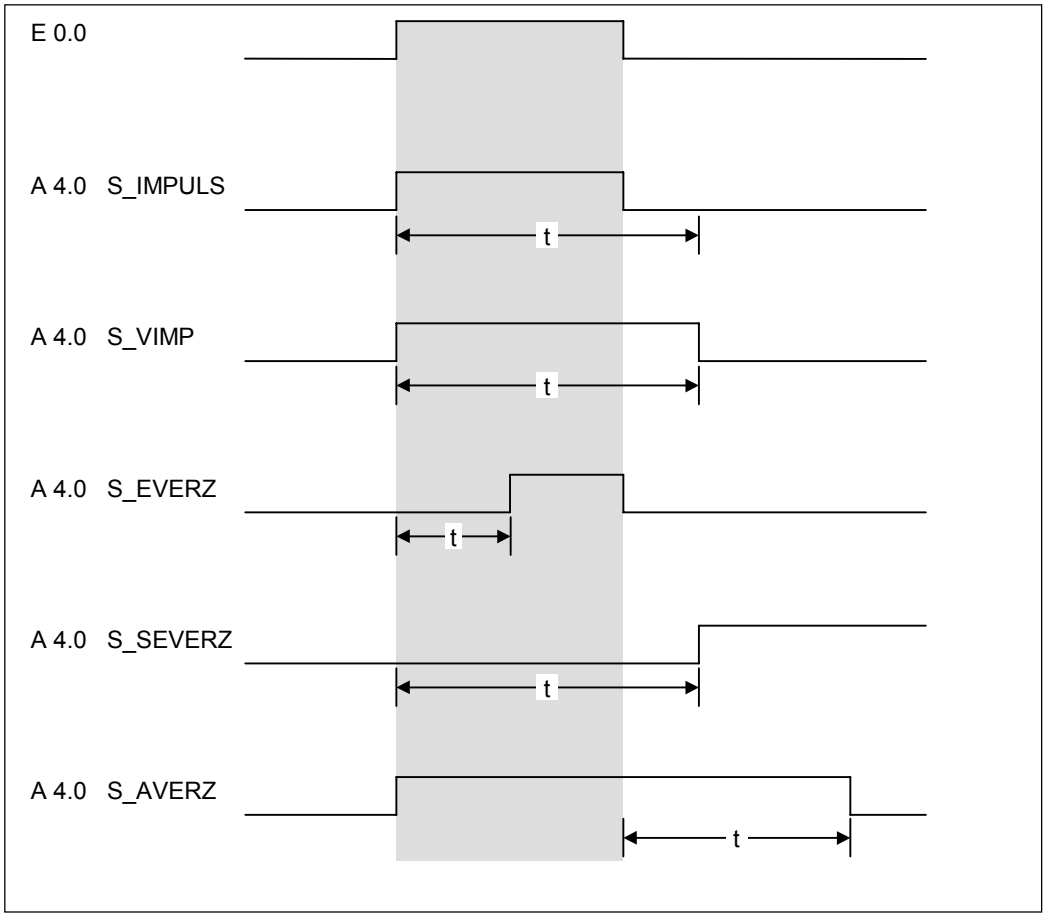


Lecture de la valeur et de la base de temps

Chaque boîte de temporisation possède deux sorties, DUAL et DEZ, pour lesquelles vous pouvez indiquer une adresse de mot. La sortie DUAL fournit la valeur de temps en format binaire, mais n'indique pas la base de temps. La sortie DEZ fournit la base de temps et la valeur de temps en format décimal codé binaire (DCB).

Choix de la temporisation correcte

La vue d'ensemble des cinq types de temporisations doit vous aider à choisir la temporisation qui répond le mieux à vos besoins.

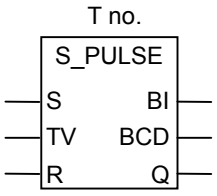


Temporisations	Description
S_IMPULS temporisation sous forme d´impulsion	La durée maximale pendant laquelle le signal de sortie reste à 1 est la même que la valeur de temps « t » programmée. Le signal de sortie reste à 1 pour une durée plus courte si le signal d'entrée passe à 0.
S_VIMP temporisation sous forme d´impulsion prolongée	Le signal de sortie reste à 1 pendant la durée programmée, quelle que soit la durée pendant laquelle le signal d'entrée reste à 1.
S_EVERZ temporisation sous forme de retard à la montée	Le signal de sortie est égal à 1 uniquement lorsque le temps programmé s'est écoulé et que le signal d'entrée est toujours à 1.
S_SEVERZ temporisation sous forme de retard à la montée mémorisé	Le signal de sortie passe de 0 à 1 uniquement lorsque le temps programmé s'est écoulé, quelle que soit la durée pendant laquelle le signal d'entrée reste à 1.
S_AVERZ temporisation sous forme de retard à la retombée	Le signal de sortie est égal à 1 lorsque le signal d'entrée est égal à 1 ou lorsque la temporisation s'exécute. La temporisation est démarrée lorsque le signal d'entrée passe de 1 à 0.

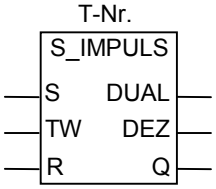
13.3 S_IMPULS : Paramétrer et démarrer une temporisation sous forme d'impulsion

Représentation

Anglaise



Allemande



Paramètre Anglaise	Paramètre Allemande	Type de données	Zone de mémoire	Description
T no.	T Nr.	TIMER	T	Numéro d'identification de la temporisation. La plage dépend de la CPU.
S	S	BOOL	E, A, M, D, L, T, Z	Entrée de démarrage
TV	TW	S5TIME	E, A, M, D, L ou constante	Valeur de temps prédéfinie (plage : 0 à 9999)
R	R	BOOL	E, A, M, D, L, T, Z	Entrée de remise à zéro
BI	DUAL	WORD	E, A, M, D, L	Valeur de temps restante (format binaire)
BCD	DEZ	WORD	E, A, M, D, L	Valeur de temps restante (format DCB)
Q	Q	BOOL	E, A, M, D, L	Etat de la temporisation

Description

L'opération **Paramétrer et démarrer une temporisation sous forme d'impulsion** démarre la temporisation précisée en cas de front montant (c'est–à–dire lorsque l'état de signal passe de 0 à 1) à l'entrée de démarrage S. Un changement d'état de signal est toujours nécessaire pour activer une temporisation. La valeur de temps indiquée à l'entrée TW s'écoule pendant le temps programmé, tant que l'état de signal à l'entrée S est égal à 1. Pendant que la temporisation s'exécute, l'interrogation à 1 de l'état de signal à la sortie Q donne 1 comme résultat. En cas de passage de 1 à 0 à l'entrée S avant que le temps n'ait expiré, la temporisation s'arrête. Dans ce cas, l'interrogation à 1 de l'état de signal à la sortie Q donne 0 comme résultat.

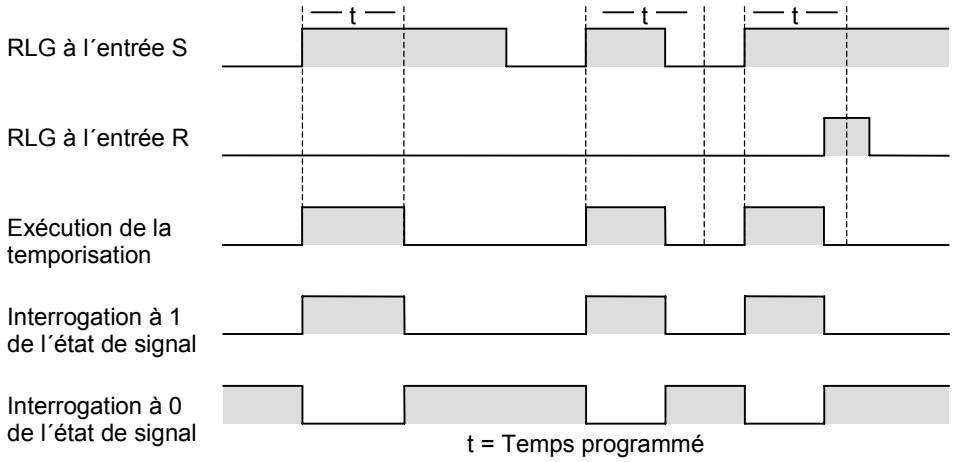
En cas de passage de 0 à 1 à l'entrée de remise à zéro R pendant que la temporisation s'écoule, cette dernière est remise à zéro. Cette transition remet aussi la valeur de temps et la base de temps à zéro. L'état de signal 1 à l'entrée R de la temporisation n'a aucun effet si la temporisation ne s'exécute pas.

La valeur de temps en cours peut être lue en format binaire à la sortie DUAL et en format décimal codé binaire à la sortie DEZ.

Voir aussi Adresse d´une temporisation en mémoire et composants d´une temporisation.

Chronogramme

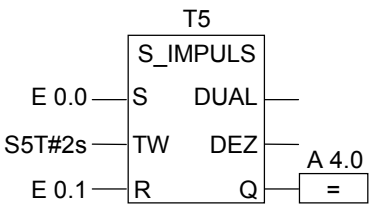
Propriétés de la temporisation sous forme d'impulsion



Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	-	-	-	-	X	X	X	1

Exemple

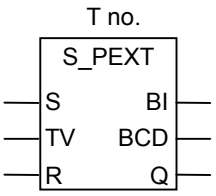


La temporisation T5 est démarrée si l'état de signal passe de 0 à 1 à l'entrée E 0.0 (front montant du RLG). Elle s'écoule avec la valeur indiquée, égale à deux secondes (2s), tant que E 0.0 est à 1. Si l'état de signal en E 0.0 passe de 1 à 0 avant que le temps n'ait expiré, la temporisation s'arrête. Si l'état de signal à l'entrée E 0.1 passe de 0 à 1 alors que la temporisation s'écoule, cette dernière est remise à zéro. L'état de signal à la sortie A 4.0 est 1 tant que la temporisation s'écoule.

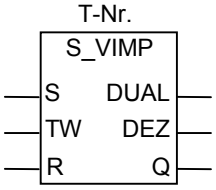
13.4 S_VIMP : Paramétrer et démarrer une temporisation sous forme d'impulsion prolongée

Représentation

Anglaise



Allemande



Paramètre Anglaise	Paramètre Allemande	Type de données	Zone de mémoire	Description
T no.	T Nr.	TIMER	T	Numéro d'identification de la temporisation. La plage dépend de la CPU.
S	S	BOOL	E, A, M, D, L, T, Z	Entrée de démarrage
TV	TW	S5TIME	E, A, M, D, L ou constante	Valeur de temps prédéfinie (plage : 0 à 9999)
R	R	BOOL	E, A, M, D, L, T, Z	Entrée de remise à zéro
BI	DUAL	WORD	E, A, M, D, L	Valeur de temps restante (format binaire)
BCD	DEZ	WORD	E, A, M, D, L	Valeur de temps restante (format DCB)
Q	Q	BOOL	E, A, M, D, L	Etat de la temporisation

Description

L'opération **Paramétrer et démarrer une temporisation sous forme d'impulsion prolongée** démarre la temporisation précisée en cas de front montant (c'est–à–dire lorsque l'état de signal passe de 0 à 1) à l'entrée de démarrage S. Un changement d'état de signal est toujours nécessaire pour activer une temporisation. La valeur de temps indiquée à l'entrée TW continue à s'écouler même si l'état de signal à l'entrée S passe à 0 avant expiration du temps. Tant que la temporisation s'écoule, l'interrogation à 1 de l'état de signal à la sortie Q donne 1 comme résultat. La temporisation est redémarrée avec la valeur de temps indiquée si l'état de signal à l'entrée S passe de 0 à 1 alors que la temporisation s'exécute.

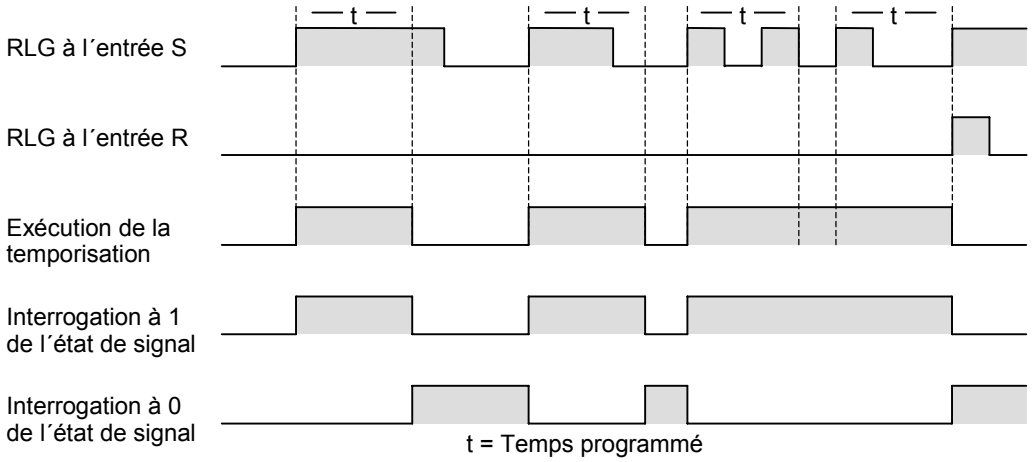
En cas de passage de 0 à 1 à l'entrée de remise à zéro (R) pendant que la temporisation s'écoule, cette dernière est remise à zéro. Cette transition remet aussi la valeur de temps et la base de temps à zéro.

La valeur de temps en cours peut être lue en format binaire à la sortie DUAL et en format décimal codé binaire à la sortie DEZ.

Voir aussi Adresse d´une temporisation en mémoire et composants d´une temporisation.

Chronogramme

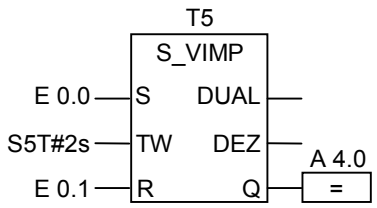
Propriétés de la temporisation sous forme d'impulsion prolongée



Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	-	-	-	-	X	X	X	1

Exemple

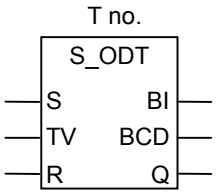


La temporisation T5 est démarrée si l'état de signal passe de 0 à 1 à l'entrée E 0.0 (front montant du RLG). Le temps de deux secondes (2s) indiqué continue à s'écouler même en cas de front descendant à l'entrée S. Si l'état de signal en E 0.0 passe de 0 à 1 avant expiration de ces deux secondes, la temporisation est redémarrée. Si l'état de signal en E 0.1 passe de 0 à 1 alors que la temporisation s'écoule, cette dernière est remise à zéro. L'état de signal à la sortie A 4.0 est 1 tant que la temporisation s'exécute.

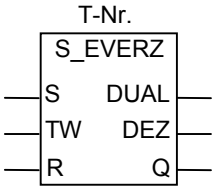
13.5 S_EVERZ : Paramétrer et démarrer une temporisation sous forme de retard à la montée

Représentation

Anglaise



Allemande



Paramètre Anglaise	Paramètre Allemande	Type de données	Zone de mémoire	Description
T no.	T Nr.	TIMER	T	Numéro d'identification de la temporisation. La plage dépend de la CPU.
S	S	BOOL	E, A, M, D, L, T, Z	Entrée de démarrage
TV	TW	S5TIME	E, A, M, D, L ou constante	Valeur de temps prédéfinie (plage : 0 à 9999)
R	R	BOOL	E, A, M, D, L, T, Z	Entrée de remise à zéro
BI	DUAL	WORD	E, A, M, D, L	Valeur de temps restante (format binaire)
BCD	DEZ	WORD	E, A, M, D, L	Valeur de temps restante (format DCB)
Q	Q	BOOL	E, A, M, D, L	Etat de la temporisation

Description

L'opération **Paramétrer et démarrer une temporisation sous forme de retard à la montée** démarre la temporisation précisée en cas de front montant (c'est–à–dire lorsque l'état de signal passe de 0 à 1) à l'entrée de démarrage S. Un changement d'état de signal est toujours nécessaire pour activer une temporisation. La valeur de temps indiquée à l'entrée TW s'écoule tant que l'état de signal à l'entrée S est à 1. L'interrogation à 1 de l'état de signal à la sortie Q donne 1 comme résultat lorsque le temps a expiré, que l'état de signal à l'entrée S est toujours 1 et que l'état de signal à l'entrée R est toujours 0. La temporisation s'arrête si l'état de signal à l'entrée S passe de 1 à 0 alors que la temporisation s'exécute. Dans ce cas, l'interrogation à 1 de l'état de signal donne toujours 0 comme résultat.

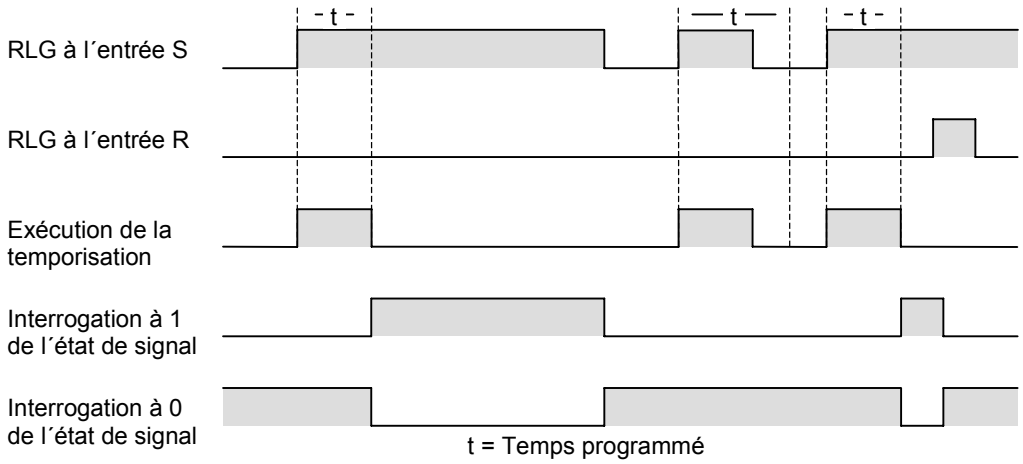
En cas de passage de 0 à 1 à l'entrée de remise à zéro (R) pendant que la temporisation s'écoule, cette dernière est remise à zéro. Cette transition remet aussi la valeur de temps et la base de temps à zéro. La temporisation est également remise à zéro si l'état de signal égale 1 à l'entrée R alors que la temporisation ne s'exécute pas.

La valeur de temps en cours peut être lue en format binaire à la sortie DUAL et en format décimal codé binaire à la sortie DEZ.

Voir aussi Adresse d´une temporisation en mémoire et composants d´une temporisation.

Chronogramme

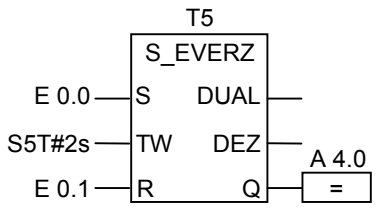
Propriétés de la temporisation sous forme de retard à la montée



Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	-	-	-	-	X	X	X	1

Exemple

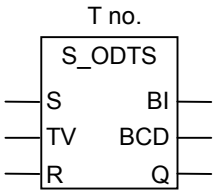


La temporisation T5 est démarrée si l'état de signal passe de 0 à 1 à l'entrée E 0.0 (front montant du RLG). Si, à l'expiration du temps de deux secondes (2s) indiqué, l'état de signal à la sortie E 0.0 est toujours 1, l'état de signal à la sortie A 4.0 est 1. Si l'état de signal en E 0.0 passe de 1 à 0, la temporisation s'arrête et A 4.0 est à 0. Si l'état de signal en E 0.0 passe de 0 à 1 alors que la temporisation s'exécute, cette dernière est redémarrée.

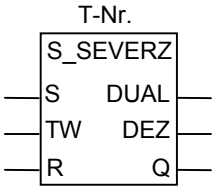
13.6 S_SEVERZ : Paramétrer et démarrer une temporisation sous forme de retard à la montée mémorisé

Représentation

Anglaise



Allemande



Paramètre Anglaise	Paramètre Allemande	Type de données	Zone de mémoire	Description
T no.	T Nr.	TIMER	T	Numéro d'identification de la temporisation. La plage dépend de la CPU.
S	S	BOOL	E, A, M, D, L, T, Z	Entrée de démarrage
TV	TW	S5TIME	E, A, M, D, L ou constante	Valeur de temps prédéfinie (plage : 0 à 9999)
R	R	BOOL	E, A, M, D, L, T, Z	Entrée de remise à zéro
BI	DUAL	WORD	E, A, M, D, L	Valeur de temps restante (format binaire)
BCD	DEZ	WORD	E, A, M, D, L	Valeur de temps restante (format DCB)
Q	Q	BOOL	E, A, M, D, L	Etat de la temporisation

Description

L'opération **Paramétrer et démarrer une temporisation sous forme de retard à la montée mémorisé** démarre la temporisation précisée en cas de front montant (c'est–à–dire lorsque l'état de signal passe de 0 à 1) à l'entrée de démarrage S. Un changement d'état de signal est toujours nécessaire pour activer une temporisation. La valeur de temps indiquée à l'entrée TW continue à s'écouler même si l'état de signal à l'entrée S passe à 0 avant que la temporisation n'ait expiré. L'interrogation à 1 de l'état de signal à la sortie Q donne 1 comme résultat lorsque le temps a expiré, quel que soit l'état de signal à l'entrée S, et lorsque l'entrée de remise à zéro (R) reste à 0. Si l'état de signal à l'entrée S passe de 0 à 1 pendant que la temporisation s'exécute, cette dernière est redémarrée avec la valeur de temps indiquée.

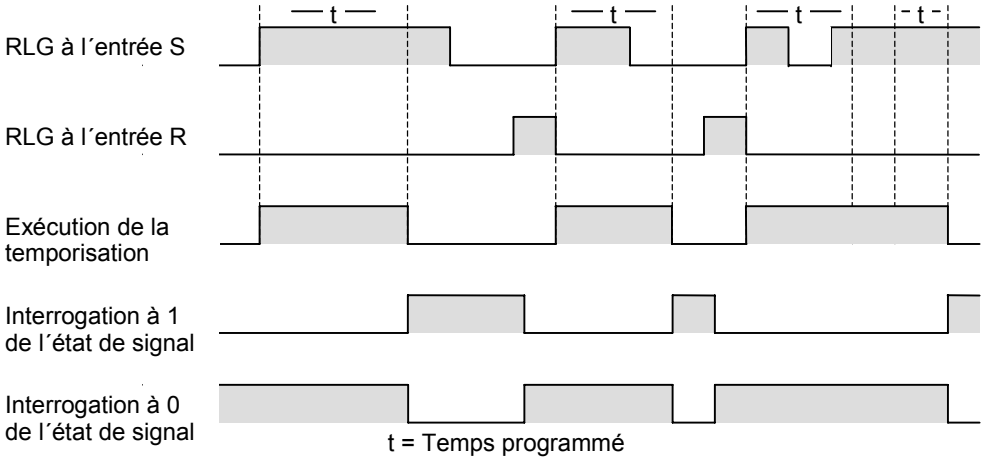
En cas de passage de 0 à 1 à l'entrée de remise à zéro (R), la temporisation est remise à zéro quel que soit le RLG à l'entrée S.

La valeur de temps en cours peut être lue en format binaire à la sortie DUAL et en format décimal codé binaire à la sortie BCD.

Voir aussi Adresse d´une temporisation en mémoire et composants d´une temporisation.

Chronogramme

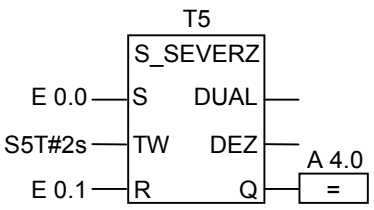
Propriétés de la temporisation sous forme de retard à la montée mémorisé



Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	-	-	-	-	X	X	X	1

Exemple

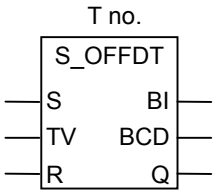


La temporisation T5 est démarrée si l'état de signal passe de 0 à 1 à l'entrée E 0.0 (front montant du RLG). Elle continue à s'écouler même si l'état de signal en E 0.0 passe de 1 à 0. Si l'état de signal à l'entrée E 0.0 passe de 0 à 1 avant expiration de la valeur indiquée, la temporisation est redémarrée. Si l'état de signal en E 0.1 passe de 0 à 1 alors que la temporisation s'écoule, cette dernière est remise à zéro. L'état de signal à la sortie A 4.0 est 1 si le temps a expiré et que E 0.1 reste à 0.

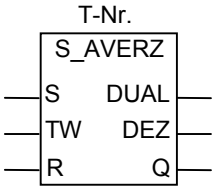
13.7 S_AVERZ : Paramétrer et démarrer une temporisation sous forme de retard à la retombée

Représentation

Anglaise



Allemande



Paramètre Anglaise	Paramètre Allemande	Type de données	Zone de mémoire	Description
T no.	T Nr.	TIMER	T	Numéro d'identification de la temporisation. La plage dépend de la CPU.
S	S	BOOL	E, A, M, D, L, T, Z	Entrée de démarrage
TV	TW	S5TIME	E, A, M, D, L ou constante	Valeur de temps prédéfinie (plage : 0 à 9999)
R	R	BOOL	E, A, M, D, L, T, Z	Entrée de remise à zéro
BI	DUAL	WORD	E, A, M, D, L	Valeur de temps restante (format binaire)
BCD	DEZ	WORD	E, A, M, D, L	Valeur de temps restante (format DCB)
Q	Q	BOOL	E, A, M, D, L	Etat de la temporisation

Description

L'opération **Paramétrer et démarrer une temporisation sous forme de retard à la retombée** démarre la temporisation précisée en cas de front descendant (c'est-à-dire lorsque l'état de signal passe de 1 à 0) à l'entrée de démarrage S. Un changement d'état de signal est toujours nécessaire pour activer une temporisation. L'interrogation à 1 de l'état de signal à la sortie Q donne 1 comme résultat lorsque l'état de signal à l'entrée S est 1 ou lorsque la temporisation s'écoule. La temporisation est remise à zéro lorsque l'état de signal à l'entrée S passe de 0 à 1 alors que la temporisation s'exécute. La temporisation n'est redémarrée que lorsque l'état de signal à l'entrée S repasse de 1 à 0.

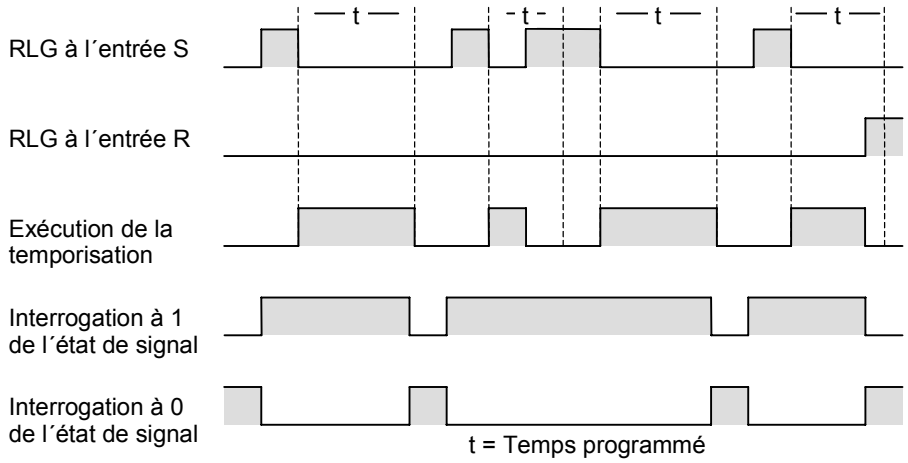
En cas de passage de 0 à 1 à l'entrée de remise à zéro (R) pendant que la temporisation s'exécute, cette dernière est remise à zéro.

La valeur de temps en cours peut être lue en format binaire à la sortie DUAL et en format décimal codé binaire à la sortie DEZ.

Voir aussi Adresse d´une temporisation en mémoire et composants d´une temporisation.

Chronogramme

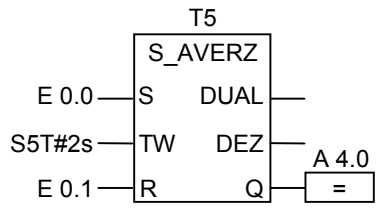
Propriétés de la temporisation sous forme de retard à la retombée



Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	-	-	-	-	X	X	X	1

Exemple

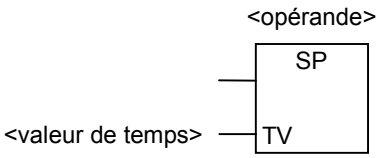


La temporisation T5 est démarrée si l'état de signal passe de 1 à 0 à l'entrée E 0.0. L'état de signal à la sortie A 4.0 est 1 lorsque l'état de signal en E 0.0 est 1 ou que la temporisation s'écoule. Si l'état de signal en E 0.1 passe de 0 à 1 alors que la temporisation s'exécute, cette dernière est remise à zéro.

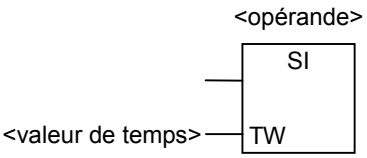
13.8 SI : Temporisation sous forme d'impulsion

Représentation

Anglaise



Allemande



Paramètre Anglaise	Paramètre Allemande	Type de données	Zone de mémoire	Description
Numéro de la temporisation	Numéro de la temporisation	TIMER	T	L'opérande indique le numéro de la temporisation à démarrer.
TV	TW	S5TIME	E, A, M, D, L ou constante	Valeur de temps (format S5TIME)

Description

L'opération **Temporisation sous forme d'impulsion** démarre la temporisation indiquée avec une valeur de temps donnée si le RLG présente un front montant (c'est–à–dire si le RLG passe de 0 à 1). La temporisation s'écoule tant que le RLG est positif. L'interrogation à 1 de l'état du signal fournit un résultat égal à 1 tant que la temporisation s'écoule. Si le RLG passe de 1 à 0 avant que le temps indiqué n'ait expiré, la temporisation s'arrête. Dans ce cas, l'interrogation à 1 de l'état de signal fournit un résultat égal à 0.

Vous ne pouvez placer la boîte **Temporisation sous forme d'impulsion** qu'à l'extrémité droite de la séquence combinatoire. Il est toutefois possible d'en utiliser plusieurs.

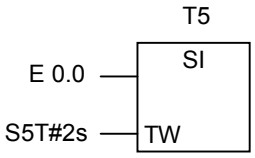
Voir aussi Adresse d'une temporisation en mémoire et composants d'une temporisation.

Mot d'état

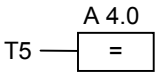
	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	-	-	-	-	0	-	-	0

Exemple

Réseau 1



Réseau 2



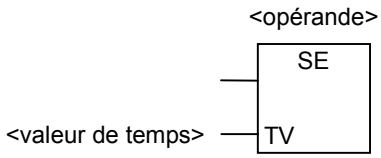
Si l'état de signal en E 0.0 passe de 0 à 1 (front montant du RLG), la temporisation T5 est démarrée. La temporisation s'écoule avec la valeur de temps précisée de 2 secondes tant que l'état de signal en E 0.0 est égal à 1. Si l'état de signal en E 0.0 passe de 1 à 0 avant expiration du temps précisé, la temporisation s'arrête.

L'état de signal à la sortie A 4.0 est 1 tant que la temporisation s'exécute.

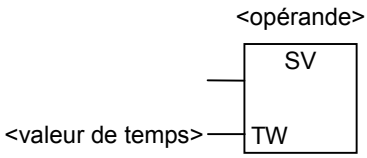
13.9 SV : Temporisation sous forme d'impulsion prolongée

Représentation

Anglaise



Allemande



Paramètre Anglaise	Paramètre Allemande	Type de données	Zone de mémoire	Description
Numéro de la temporisation	Numéro de la temporisation	TIMER	T	L'opérande indique le numéro de la temporisation à démarrer.
TV	TW	S5TIME	E, A, M, D, L ou constante	Valeur de temps (format S5TIME)

Description

L'opération **Temporisation sous forme d'impulsion prolongée** démarre la temporisation indiquée avec une valeur de temps donnée si le RLG présente un front montant (c'est–à–dire si le RLG passe de 0 à 1). La temporisation continue à s'écouler même si le RLG passe à 0 avant que le temps précisé n'ait expiré. L'interrogation à 1 de l'état du signal fournit un résultat égal à 1 tant que la temporisation s'écoule. La temporisation est redémarrée (redéclenchée) avec le temps indiqué si le RLG repasse de 0 à 1 alors que la temporisation s'écoule encore.

Vous ne pouvez placer la boîte **Temporisation sous forme d'impulsion prolongée** qu'à l'extrémité droite de la séquence combinatoire. Il est toutefois possible d'en utiliser plusieurs.

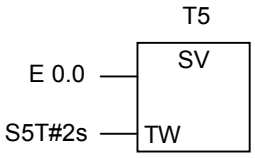
Voir aussi Adresse d'une temporisation en mémoire et composants d'une temporisation.

Mot d'état

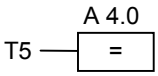
	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	-	-	-	-	0	-	-	0

Exemple

Réseau 1



Réseau 2



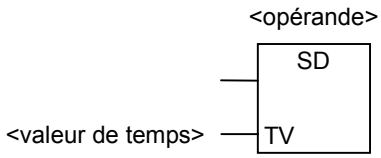
Si l'état de signal en E 0.0 passe de 0 à 1 (front montant du RLG), la temporisation T5 est démarrée. La temporisation continue à s'écouler même en présence d'un front descendant du RLG. Si l'état de signal en E 0.0 repasse de 0 à 1 avant expiration du temps précisé, la temporisation est redéclenchée.

L'état de signal à la sortie A 4.0 est 1 tant que la temporisation s'exécute.

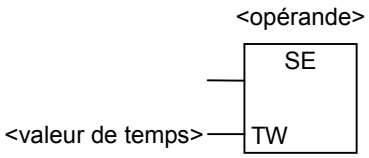
13.10 SE : Temporisation sous forme de retard à la montée

Représentation

Anglaise



Allemande



Paramètre Anglaise	Paramètre Allemande	Type de données	Zone de mémoire	Description
Numéro de la temporisation	Numéro de la temporisation	TIMER	T	L'opérande indique le numéro de la temporisation à démarrer.
TV	TW	S5TIME	E, A, M, D, L ou constante	Valeur de temps (format S5TIME)

Description

L'opération **Temporisation sous forme de retard à la montée** démarre la temporisation indiquée si le RLG présente un front montant (c'est–à–dire si le RLG passe de 0 à 1). L'interrogation à 1 de l'état du signal fournit un résultat égal à 1 lorsque le temps indiqué s'est écoulé sans erreur et que le RLG est toujours égal à 1. Si le RLG passe de 1 à 0 alors que la temporisation s'écoule, la temporisation est arrêtée. Dans ce cas, l'interrogation à 1 de l'état de signal fournit toujours un résultat égal à 0.

Vous ne pouvez placer la boîte **Temporisation sous forme de retard à la montée** qu'à l'extrémité droite de la séquence combinatoire. Il est toutefois possible d'en utiliser plusieurs.

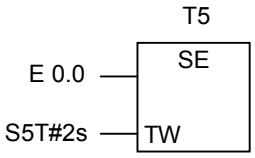
Voir aussi Adresse d'une temporisation en mémoire et composants d'une temporisation.

Mot d'état

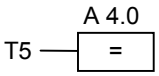
	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	-	-	-	-	0	-	-	0

Exemple

Réseau 1



Réseau 2

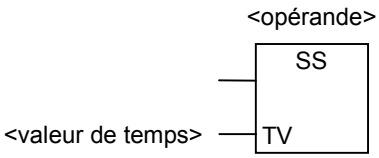


Si l'état de signal en E 0.0 passe de 0 à 1 (front montant du RLG), la temporisation T5 est démarrée. Si le temps expire et que l'état de signal en E 0.0 est toujours 1, la sortie A 4.0 est mise à 1. Si l'état de signal en E 0.0 passe de 1 à 0, la temporisation est arrêtée.

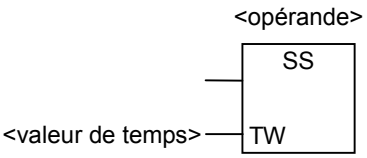
13.11 SS : Temporisation sous forme de retard à la montée mémorisé

Représentation

Anglaise



Allemande



Paramètre Anglaise	Paramètre Allemande	Type de données	Zone de mémoire	Description
Numéro de la temporisation	Numéro de la temporisation	TIMER	T	L'opérande indique le numéro de la temporisation à démarrer.
TV	TW	S5TIME	E, A, M, D, L ou constante	Valeur de temps (format S5TIME)

Description

L'opération **Temporisation sous forme de retard à la montée mémorisé** démarre la temporisation indiquée si le RLG présente un front montant (c'est–à–dire si le RLG passe de 0 à 1). La temporisation continue à s'écouler même si le RLG passe à 0 avant que le temps n'ait expiré. L'interrogation à 1 de l'état du signal fournit un résultat égal à 1 lorsque le temps indiqué a expiré, quel que soit le RLG. Si le RLG passe de 0 à 1 alors que la temporisation s'exécute, la temporisation est redémarrée (redéclenchée) avec le temps indiqué.

Vous ne pouvez placer la boîte **Temporisation sous forme de retard à la montée mémorisé** qu'à l'extrémité droite de la séquence combinatoire. Il est toutefois possible d'en utiliser plusieurs.

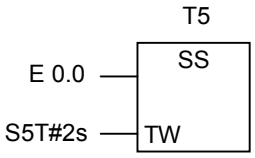
Voir aussi Adresse d'une temporisation en mémoire et composants d'une temporisation.

Mot d'état

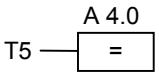
	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	-	-	-	-	0	-	-	0

Exemple

Réseau 1



Réseau 2



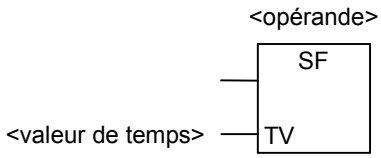
Si l'état de signal en E 0.0 passe de 0 à 1 (front montant du RLG), la temporisation T5 est démarrée. La temporisation continue à s'écouler même si l'état de signal passe de 1 à 0 à l'entrée E 0.0. Si l'état de signal en E 0.0 passe de 0 à 1 avant que le temps n'ait expiré, la temporisation est redéclenchée.

L'état de signal de la sortie A 4.0 est à 1 si le temps a expiré.

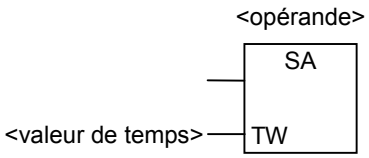
13.12 SA : Temporisation sous forme de retard à la retombée

Représentation

Anglaise



Allemande



Paramètre Anglaise	Paramètre Allemande	Type de données	Zone de mémoire	Description
Numéro de la temporisation	Numéro de la temporisation	TIMER	T	L'opérande indique le numéro de la temporisation à démarrer.
TV	TW	S5TIME	E, A, M, D, L ou constante	Valeur de temps (format S5TIME)

Description

L'opération **Temporisation sous forme de retard à la retombée** démarre la temporisation indiquée si le RLG présente un front descendant (c'est–à–dire si le RLG passe de 1 à 0). L'interrogation à 1 de l'état du signal fournit un résultat égal à 1 lorsque le RLG est égal à 1 ou que la temporisation s'écoule. La temporisation est mise à 0 lorsque le RLG passe de 0 à 1 alors que la temporisation s'exécute. La temporisation n'est redémarrée que lorsque le RLG repasse de 1 à 0.

Vous ne pouvez placer la boîte **Temporisation sous forme de retard à la retombée** qu'à l'extrémité droite de la séquence combinatoire. Il est toutefois possible d'en utiliser plusieurs.

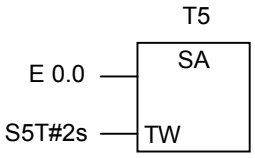
Voir aussi Adresse d'une temporisation en mémoire et composants d'une temporisation.

Mot d'état

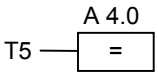
	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	-	-	-	-	0	-	-	0

Exemple

Réseau 1



Réseau 2



Si l'état de signal en E 0.0 passe de 1 à 0, la temporisation T5 est démarrée.

Si l'état de signal en E 0.0 passe de 0 à 1, la temporisation T5 est mise à 0.

L'état de signal de la sortie A 4.0 est à 1 lorsque l'état de signal est 1 à l'entrée E 0.0 ou que la temporisation s'exécute.

14 Opérations combinatoires sur mots

14.1 Vue d'ensemble des opérations combinatoires sur mots

Description

Les opérations combinatoires sur mots combinent deux mots (16 bits) ou deux doubles mots (32 bits) indiqués dans les entrées IN1 et IN2, selon les combinaisons booléennes. Ces opérations sont activées si l'état de signal est 1 à l'entrée de validation EN. Les valeurs sont interprétées comme profils binaires purs. Le résultat est rangé dans la sortie OUT. L'état de signal de ENO est identique à celui de EN.

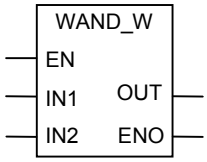
Si le résultat à la sortie OUT est différent de 0, le bit BI1 du mot d'état est mis à 1. Si le résultat à la sortie OUT égale 0, le bit BI1 du mot d'état est mis à 0.

Vous disposez des opérations combinatoires sur mots suivantes :

- WAND_W : ET mot
- WOR_W : OU mot
- WXOR_W : OU exclusif mot
- WAND_DW : ET double mot
- WOR_DW : OU double mot
- WXOR_DW : OU exclusif double mot

14.2 WAND_W : ET mot

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN1	WORD	E, A, M, D, L ou constante	Première valeur de la combinaison
IN2	WORD	E, A, M, D, L ou constante	Seconde valeur de la combinaison
OUT	WORD	E, A, M, D, L	Résultat de la combinaison
ENO	BOOL	E, A, M, D, L	Sortie de validation

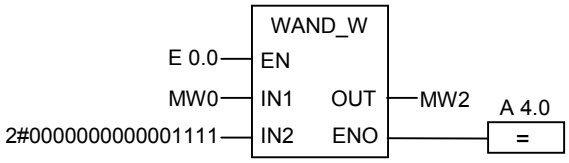
Description

L'opération **ET mot** est activée si l'état de signal est 1 à l'entrée de validation EN. Cette opération combine, bit par bit selon la table de vérité ET, les deux mots indiqués dans les entrées IN1 et IN2. Ces valeurs sont interprétées comme profils binaires purs. Le résultat est rangé dans la sortie OUT. L'état de signal de ENO est identique à celui de EN.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	1	X	0	0	-	X	1	1	1

Exemple



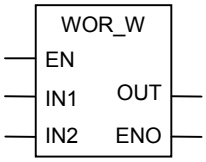
L'opération est activée si l'état de signal est 1 à l'entrée E 0.0. Seuls les bits 0 à 3 sont significatifs ; les autres bits de MW0 sont masqués :

IN1 = 0101010101010101
IN2 = 0000000000001111
OUT = 0000000000000101

La sortie A 4.0 est mise à 1 si la combinaison est exécutée.

14.3 WOR_W : OU mot

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN1	WORD	E, A, M, D, L ou constante	Première valeur de la combinaison
IN2	WORD	E, A, M, D, L ou constante	Seconde valeur de la combinaison
OUT	WORD	E, A, M, D, L	Résultat de la combinaison
ENO	BOOL	E, A, M, D, L	Sortie de validation

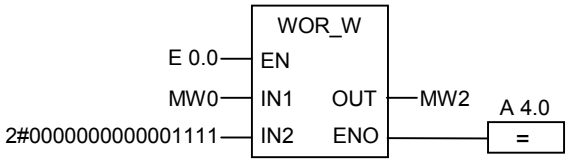
Description

L'opération **OU mot** est activée si l'état de signal est 1 à l'entrée de validation EN. Cette opération combine, bit par bit selon la table de vérité OU, les deux mots indiqués dans les entrées IN1 et IN2. Ces valeurs sont interprétées comme profils binaires purs. Le résultat est rangé dans la sortie OUT. L'état de signal de ENO est identique à celui de EN.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	1	X	0	0	-	X	1	1	1

Exemple



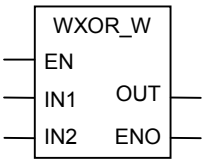
L'opération est activée si l'état de signal est 1 à l'entrée E 0.0. Les bits dans MW0 et ceux dans la constante sont combinés selon OU. Les bits 0 à 3 sont mis à 1 ; les autres bits de MW0 restent inchangés dans MW2.

IN1 = 0101010101010101
IN2 = 0000000000001111
OUT = 0101010101011111

La sortie A 4.0 est mise à 1 si la combinaison est exécutée.

14.4 WXOR_W : OU exclusif mot

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN1	WORD	E, A, M, D, L ou constante	Première valeur de la combinaison
IN2	WORD	E, A, M, D, L ou constante	Seconde valeur de la combinaison
OUT	WORD	E, A, M, D, L	Résultat de la combinaison
ENO	BOOL	E, A, M, D, L	Sortie de validation

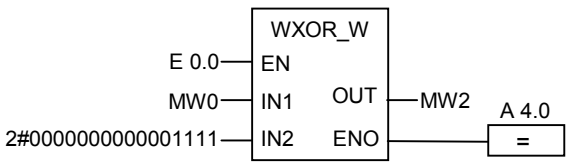
Description

L'opération **OU exclusif mot** est activée si l'état de signal est 1 à l'entrée de va-lidation EN. Cette opération combine, bit par bit selon la table de vérité OU exclusif, les deux mots indiqués dans les entrées IN1 et IN2. Ces valeurs sont interprétées comme profils binaires purs. Le résultat est rangé dans la sortie OUT. L'état de signal de ENO est identique à celui de EN.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	1	X	0	0	-	X	1	1	1

Exemple



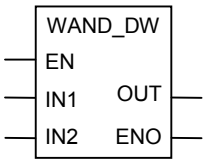
L'opération est activée si l'état de signal est 1 à l'entrée E 0.0.

IN1 = 0101010101010101
IN2 = 0000000000001111
OUT = 0101010101011010

La sortie A 4.0 est mise à 1 si la combinaison est exécutée.

14.5 WAND_DW : ET double mot

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN1	DWORD	E, A, M, D, L ou constante	Première valeur de la combinaison
IN2	DWORD	E, A, M, D, L ou constante	Seconde valeur de la combinaison
OUT	DWORD	E, A, M, D, L	Résultat de la combinaison
ENO	BOOL	E, A, M, D, L	Sortie de validation

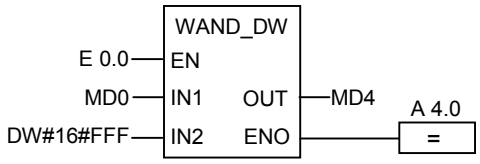
Description

L'opération **ET double mot** est activée si l'état de signal est 1 à l'entrée de validation EN. Cette opération combine, bit par bit selon la table de vérité ET, les deux doubles mots indiqués dans les entrées IN1 et IN2. Ces valeurs sont interprétées comme profils binaires purs. Le résultat est rangé dans la sortie OUT. L'état de signal de ENO est identique à celui de EN.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	1	X	0	0	-	X	1	1	1

Exemple



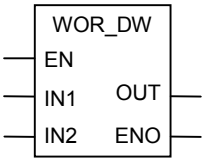
L'opération est activée si l'état de signal est 1 à l'entrée E 0.0. Seuls les bits 0 à 11 sont significatifs ; les autres bits de MD0 sont masqués :

IN1 = 01010101010101010101010101010101
IN2 = 00000000000000000000111111111111
OUT = 00000000000000000000010101010101

La sortie A 4.0 est mise à 1 si la combinaison est exécutée.

14.6 WOR_DW : OU double mot

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN1	DWORD	E, A, M, D, L ou constante	Première valeur de la combinaison
IN2	DWORD	E, A, M, D, L ou constante	Seconde valeur de la combinaison
OUT	DWORD	E, A, M, D, L	Résultat de la combinaison
ENO	BOOL	E, A, M, D, L	Sortie de validation

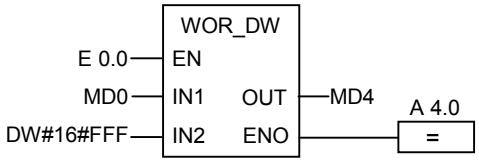
Description

L'opération **OU double mot** est activée si l'état de signal est 1 à l'entrée de validation EN. Cette opération combine, bit par bit selon la table de vérité OU, les deux doubles mots indiqués dans les entrées IN1 et IN2. Ces valeurs sont interprétées comme profils binaires purs. Le résultat est rangé dans la sortie OUT. L'état de signal de ENO est identique à celui de EN.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	1	X	0	0	-	X	1	1	1

Exemple



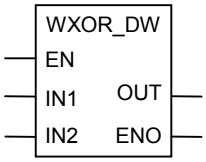
L'opération est activée si l'état de signal est 1 à l'entrée E 0.0. Les bits dans MW0 et ceux dans la constante sont combinés selon OU. Les bits 0 à 11 sont mis à 1 ; les autres bits de MW0 restent inchangés dans MW2.

IN1 = 0101010101010101 0101010101010101
IN2 = 0000000000000000 0000111111111111
OUT = 0101010101010101 0101111111111111

La sortie A 4.0 est mise à 1 si la combinaison est exécutée.

14.7 WXOR_DW : OU exclusif double mot

Représentation



Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, D, L, T, Z	Entrée de validation
IN1	DWORD	E, A, M, D, L ou constante	Première valeur de la combinaison
IN2	DWORD	E, A, M, D, L ou constante	Seconde valeur de la combinaison
OUT	DWORD	E, A, M, D, L	Résultat de la combinaison
ENO	BOOL	E, A, M, D, L	Sortie de validation

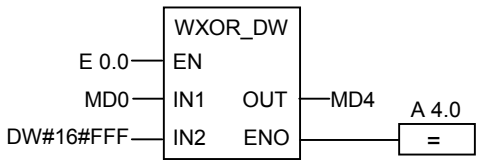
Description

L'opération **OU exclusif double mot** est activée si l'état de signal est 1 à l'entrée de validation EN. Cette opération combine, bit par bit selon la table de vérité OU exclusif, les deux doubles mots indiqués dans les entrées IN1 et IN2. Ces valeurs sont interprétées comme profils binaires purs. Le résultat est rangé dans la sortie OUT. L'état de signal de ENO est identique à celui de EN.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	1	X	0	0	-	X	1	1	1

Exemple



L'opération est activée si l'état de signal est 1 à l'entrée E 0.0.

IN1 = 0101010101010101 0101010101010101
IN2 = 0000000000000000 0000111111111111
OUT = 0101010101010101 0101101010101010

La sortie A 4.0 est mise à 1 si la combinaison est exécutée.

A Présentation de toutes les opérations LOG

A.1 Opérations LOG classées d'après les abréviations allemandes (SIMATIC)

Abréviations allemandes	Abréviations anglaises	Catalogue des éléments du programme	Description
&	&	Combinaison sur bits	Porte ET
>=1	>=1	Combinaison sur bits	Porte OU
=	=	Combinaison sur bits	Affectation
#	#	Combinaison sur bits	Connecteur
---|	---|	Combinaison sur bits	Insérer entrée binaire
---o|	---o|	Combinaison sur bits	Inverser entrée binaire
==0	==0	Bits d'état	Bits de résultat
>0	>0	Bits d'état	Bits de résultat
>=0	>=0	Bits d'état	Bits de résultat
<0	<0	Bits d'état	Bits de résultat
<=0	<=0	Bits d'état	Bits de résultat
<>0	<>0	Bits d'état	Bits de résultat
ABS	ABS	Fonction sur nombres à virgule flottante	Valeur absolue d'un nombre réel
ACOS	ACOS	Fonction sur nombres à virgule flottante	Fonctions trigonométriques d'angles sous forme de nombres réels
ADD_DI	ADD_DI	Fonction sur nombres entiers	Additionner entier de 32 bits
ADD_I	ADD_I	Fonction sur nombres entiers	Additionner entier de 16 bits
ADD_R	ADD_R	Fonction sur nombres à virgule flottante	Additionner nombres réels
ASIN	ASIN	Fonction sur nombres à virgule flottante	Fonctions trigonométriques d'angles sous forme de nombres réels
ATAN	ATAN	Fonction sur nombres à virgule flottante	Fonctions trigonométriques d'angles sous forme de nombres réels
BCD_DI	BCD_DI	Conversion	Convertir nombre DCB en entier de 32 bits
BCD_I	BCD_I	Conversion	Convertir nombre DCB en entier de 16 bits
BIE	BR	Bits d'état	Bit d'anomalie "Registre RB"
CALL	CALL	Gestion d'exécution de programmes	Appeler FC/SFC sans paramètre
CALL_FB	CALL_FB	Gestion d'exécution de programmes	CALL_FB : Appeler FB
CALL_FC	CALL_FC	Gestion d'exécution de programmes	Appeler FC (CALL_FC)
CALL_SFB	CALL_SFB	Gestion d'exécution de programmes	Appeler SFB (CALL_SFB)
CALL_SFC	CALL_SFC	Gestion d'exécution de programmes	Appeler SFC (CALL_SFC)

Abréviations allemandes	Abréviations anglaises	Catalogue des éléments du programme	Description
CEIL	CEIL	Conversion	Convertir nombre réel en entier supérieur le plus proche
CMP >=D	CMP >=D	Comparaison	Comparer entiers de 32 bits
CMP >=I	CMP >=I	Comparaison	Comparer entiers de 16 bits
CMP >=R	CMP >=R	Comparaison	Comparer nombres réels
COS	COS	Fonction sur nombres à virgule flottante	Fonctions trigonométriques d'angles sous forme de nombres réels
DI_BCD	DI_BCD	Conversion	Convertir entier de 32 bits en nombre DCB
DI_R	DI_R	Conversion	Convertir entier de 32 bits en nombre réel
DIV_DI	DIV_DI	Fonction sur nombres entiers	Diviser entiers de 32 bits
DIV_I	DIV_I	Fonction sur nombres entiers	Diviser entiers de 16 bits
DIV_R	DIV_R	Fonction sur nombres à virgule flottante	Diviser nombres réels
EXP	EXP	Fonction sur nombres à virgule flottante	Valeur exponentielle d'un nombre réel
FLOOR	FLOOR	Conversion	Convertir nombre réel en entier inférieur le plus proche
I_BCD	I_BCD	Conversion	Convertir entier de 16 bits en nombre DCB
I_DI	I_DI	Conversion	Convertir entier de 16 bits en entier de 32 bits
INV_I	INV_I	Conversion	Complément à 1 d'entier de 16 bits
INV_DI	INV_DI	Conversion	Complément à 1 d'entier de 32 bits
JMP	JMP	Sauts	Saut inconditionnel
JMP	JMP	Sauts	Saut si 1 (conditionnel)
JMPN	JMPN	Sauts	Saut si 0 (conditionnel)
LABEL	LABEL	Sauts	Repère de saut
LN	LN	Fonction sur nombres à virgule flottante	Logarithme naturel d'un nombre réel
MCR>	MCR>	Gestion d'exécution de programmes	Désactiver relais de masquage
MCR<	MCR<	Gestion d'exécution de programmes	Activer relais de masquage
MCRA	MCRA	Gestion d'exécution de programmes	Relais de masquage en fonction
MCRD	MCRD	Gestion d'exécution de programmes	Relais de masquage en fonction
MOD_DI	MOD_DI	Fonction sur nombres entiers	Reste de division (32 bits)
MOVE	MOVE	Transfert	Affecter valeur
MUL_DI	MUL_DI	Fonction sur nombres entiers	Multiplier entiers de 32 bits
MUL_I	MUL_I	Fonction sur nombres entiers	Multiplier entiers de 16 bits
MUL_R	MUL_R	Fonction sur nombres à virgule flottante	Multiplier nombres réels
N	N	Combinaison sur bits	Détecter front descendant
NEG	NEG	Combinaison sur bits	Détecter front montant
NEG_DI	NEG_DI	Conversion	Complément à deux d'entier de 32 bits
NEG_I	NEG_I	Conversion	Complément à deux d'entier de 16 bits
NEG_R	NEG_R	Conversion	Inverser le signe d'un nombre réel
OPN	OPN	Appel de DB	Ouvrir bloc de données

Abréviations allemandes	Abréviations anglaises	Catalogue des éléments du programme	Description
OS	OS	Bits d'état	Bit d'anomalie "Débordement mémorisé"
OV	OV	Bits d'état	Bit d'anomalie "Débordement
P	P	Combinaison sur bits	Détecter front montant
POS	POS	Combinaison sur bits	Détecter front montant de signal
R	R	Combinaison sur bits	Mettre à 0
RET	RET	Gestion d'exécution de programmes	Retour
ROL_DW	ROL_DW	Décalage/rotation	Rotation vers la gauche d'un double mot
ROUND	ROUND	Conversion	Arrondir à entier de 32 bits
ROR_DW	ROR_DW	Décalage/rotation	Rotation vers la droite d'un double mot
RS	RS	Combinaison sur bits	Bascule mise à 0, mise à 1
S	S	Combinaison sur bits	Mettre à 1
SA	SF	Temporisations	Paramétrer et démarrer une temporisation sous forme de retard à la retombée
SAVE	SAVE	Combinaison sur bits	Sauvegarder RLG dans RB
S_AVERZ	S_OFFDT	Temporisations	Paramétrer et démarrer une temporisation sous forme de retard à la retombée
SE	SD	Temporisations	Paramétrer et démarrer une temporisation sous forme de retard à la montée
S_EVERZ	S_ODT	Temporisations	Paramétrer et démarrer une temporisation sous forme de retard à la montée
SHL_DW	SHL_DW	Décalage/rotation	Décalage vers la gauche d'un double mot
SHL_W	SHL_W	Décalage/rotation	Décalage vers la gauche d'un mot
SHR_DI	SHR_DI	Décalage/rotation	Décalage vers la droite d'un entier de 32 bits
SHR_DW	SHR_DW	Décalage/rotation	Décalage vers la droite d'un double mot
SHR_I	SHR_I	Décalage/rotation	Décalage vers la droite d'un entier de 16 bits
SHR_W	SHR_W	Décalage/rotation	Décalage vers la droite d'un mot
SI	SP	Temporisations	Temporisation sous forme d'impulsion
S_IMPULS	S_PULSE	Temporisations	Paramétrer et démarrer une temporisation sous forme d'impulsion
SIN	SIN	Fonction sur nombres à virgule flottante	Fonctions trigonométriques d'angles sous forme de nombres réels
SQR	SQR	Fonction sur nombres à virgule flottante	Carré d'un nombre réel
SQRT	SQRT	Fonction sur nombres à virgule flottante	Racine carrée d'un nombre réel
SR	SR	Combinaison sur bits	Bascule mise à 1, mise à 0
SS	SS	Temporisations	Temporisation sous forme de retard à la montée mémorisé
S_SEVERZ	S_ODTS	Temporisations	Paramétrer et démarrer une temporisation sous forme de retard à la montée mémorisé
SUB_DI	SUB_DI	Fonction sur nombres entiers	Soustraire entiers de 32 bits
SUB_I	SUB_I	Fonction sur nombres entiers	Soustraire entiers de 16 bits
SUB_R	SUB_R	Fonction sur nombres à virgule flottante	Soustraire nombres réels
SV	SE	Temporisations	Temporisation sous forme d'impulsion prolongée
S_VIMP	S_PEXT	Temporisations	Paramétrer et démarrer une temporisation sous forme d'impulsion prolongée
SZ	SC	Compteurs	Initialiser compteur

Abréviations allemandes	Abréviations anglaises	Catalogue des éléments du programme	Description
TAN	TAN	Fonction sur nombres à virgule flottante	Fonctions trigonométriques d'angles sous forme de nombres réels
TRUNC	TRUNC	Conversion	Tronquer à la partie entière (32 bits)
UO	UO	Bits d'état	Bit d'anomalie "Opération illicite"
WAND_DW	WAND_DW	Combinaison sur mots	ET double mot
WAND_W	WAND_W	Combinaison sur mots	ET mot
WOR_DW	WOR_DW	Combinaison sur mots	OU double mot
WOR_W	WOR_W	Combinaison sur mots	OU mot
WXOR_DW	WXOR_DW	Combinaison sur mots	OU exclusif double mot
WXOR_W	WXOR_W	Combinaison sur mots	OU exclusif mot
XOR	XOR	Combinaison sur bits	Combinaison OU exclusif
ZAEHLER	S_CUD	Compteurs	Paramétrage et compteur incrémental/décrémental
ZR	CD	Compteurs	Décrémenter
Z_RUECK	S_CD	Compteurs	Paramétrage et compteur décrémental
ZV	CU	Compteurs	Incrémenter
Z_VORW	S_CU	Compteurs	Paramétrage et compteur incrémental

A.2 Opérations LOG classées d'après les abréviations anglaises (internationales)

Abréviations allemandes	Abréviations anglaises	Catalogue des éléments du programme	Description
&	&	Combinaison sur bits	Porte ET
>=1	>=1	Combinaison sur bits	Porte OU
=	=	Combinaison sur bits	Affectation
#	#	Combinaison sur bits	Connecteur
---|	---|	Combinaison sur bits	Insérer entrée binaire
---o|	---o|	Combinaison sur bits	Inverser entrée binaire
==0	==0	Bits d'état	Bits de résultat
>0	>0	Bits d'état	Bits de résultat
>=0	>=0	Bits d'état	Bits de résultat
<0	<0	Bits d'état	Bits de résultat
<=0	<=0	Bits d'état	Bits de résultat
<>0	<>0	Bits d'état	Bits de résultat
ABS	ABS	Fonction sur nombres à virgule flottante	Valeur absolue d'un nombre réel
ACOS	ACOS	Fonction sur nombres à virgule flottante	Fonctions trigonométriques d'angles sous forme de nombres réels
ADD_DI	ADD_DI	Fonction sur nombres entiers	Additionner entier de 32 bits
ADD_I	ADD_I	Fonction sur nombres entiers	Additionner entier de 16 bits
ADD_R	ADD_R	Fonction sur nombres à virgule flottante	Additionner nombres réels
ASIN	ASIN	Fonction sur nombres à virgule flottante	Fonctions trigonométriques d'angles sous forme de nombres réels
ATAN	ATAN	Fonction sur nombres à virgule flottante	Fonctions trigonométriques d'angles sous forme de nombres réels
BCD_DI	BCD_DI	Conversion	Convertir nombre DCB en entier de 32 bits
BCD_I	BCD_I	Conversion	Convertir nombre DCB en entier de 16 bits
BR	BIE	Bits d'état	Bit d'anomalie "Registre RB"
CALL	CALL	Gestion d'exécution de programmes	Appeler FC/SFC sans paramètre
CALL_FB	CALL_FB	Gestion d'exécution de programmes	CALL_FB : Appeler FB
CALL_FC	CALL_FC	Gestion d'exécution de programmes	Appeler FC (CALL_FC)
CALL_SFB	CALL_SFB	Gestion d'exécution de programmes	Appeler SFB (CALL_SFB)
CALL_SFC	CALL_SFC	Gestion d'exécution de programmes	Appeler SFC (CALL_SFC)
CD	ZR	Compteurs	Décrémenter
CEIL	CEIL	Conversion	Convertir nombre réel en entier supérieur le plus proche
CMP >=D	CMP >=D	Comparaison	Comparer entiers de 32 bits
CMP >=I	CMP >=I	Comparaison	Comparer entiers de 16 bits
CMP >=R	CMP >=R	Comparaison	Comparer nombres réels
COS	COS	Fonction sur nombres à virgule flottante	Fonctions trigonométriques d'angles sous forme de nombres réels
CU	ZV	Compteurs	Incrémenter
DI_BCD	DI_BCD	Conversion	Convertir entier de 32 bits en nombre DCB

Abréviations allemandes	Abréviations anglaises	Catalogue des éléments du programme	Description
DI_R	DI_R	Conversion	Convertir entier de 32 bits en nombre réel
DIV_DI	DIV_DI	Fonction sur nombres entiers	Diviser entiers de 32 bits
DIV_I	DIV_I	Fonction sur nombres entiers	Diviser entiers de 16 bits
DIV_R	DIV_R	Fonction sur nombres à virgule flottante	Diviser nombres réels
EXP	EXP	Fonction sur nombres à virgule flottante	Valeur exponentielle d'un nombre réel
FLOOR	FLOOR	Conversion	Convertir nombre réel en entier inférieur le plus proche
I_BCD	I_BCD	Conversion	Convertir entier de 16 bits en nombre DCB
I_DI	I_DI	Conversion	Convertir entier de 16 bits en entier de 32 bits
INV_I	INV_I	Conversion	Complément à 1 d'entier de 16 bits
INV_DI	INV_DI	Conversion	Complément à 1 d'entier de 32 bits
JMP	JMP	Sauts	Saut inconditionnel
JMP	JMP	Sauts	Saut si 1 (conditionnel)
JMPN	JMPN	Sauts	Saut si 0 (conditionnel)
LABEL	LABEL	Sauts	Repère de saut
LN	LN	Fonction sur nombres à virgule flottante	Logarithme naturel d'un nombre réel
MCR>	MCR>	Gestion d'exécution de programmes	Désactiver relais de masquage
MCR<	MCR<	Gestion d'exécution de programmes	Activer relais de masquage
MCRA	MCRA	Gestion d'exécution de programmes	Relais de masquage en fonction
MCRD	MCRD	Gestion d'exécution de programmes	Relais de masquage en fonction
MOD_DI	MOD_DI	Fonction sur nombres entiers	Reste de division (32 bits)
MOVE	MOVE	Transfert	Affecter valeur
MUL_DI	MUL_DI	Fonction sur nombres entiers	Multiplier entiers de 32 bits
MUL_I	MUL_I	Fonction sur nombres entiers	Multiplier entiers de 16 bits
MUL_R	MUL_R	Fonction sur nombres à virgule flottante	Multiplier nombres réels
N	N	Combinaison sur bits	Détecter front descendant
NEG	NEG	Combinaison sur bits	Détecter front montant
NEG_DI	NEG_DI	Conversion	Complément à deux d'entier de 32 bits
NEG_I	NEG_I	Conversion	Complément à deux d'entier de 16 bits
NEG_R	NEG_R	Conversion	Inverser le signe d'un nombre réel
OPN	OPN	Appel de DB	Ouvrir bloc de données
OS	OS	Bits d'état	Bit d'anomalie "Débordement mémorisé"
OV	OV	Bits d'état	Bit d'anomalie "Débordement
P	P	Combinaison sur bits	Détecter front montant
POS	POS	Combinaison sur bits	Détecter front montant de signal
R	R	Combinaison sur bits	Mettre à 0
RET	RET	Gestion d'exécution de programmes	Retour
ROL_DW	ROL_DW	Décalage/rotation	Rotation vers la gauche d'un double mot
ROUND	ROUND	Conversion	Arrondir à entier de 32 bits
ROR_DW	ROR_DW	Décalage/rotation	Rotation vers la droite d'un double mot
RS	RS	Combinaison sur bits	Bascule mise à 0, mise à 1
S	S	Combinaison sur bits	Mettre à 1

Abréviations allemandes	Abréviations anglaises	Catalogue des éléments du programme	Description
SAVE	SAVE	Combinaison sur bits	Sauvegarder RLG dans RB
SC	SZ	Compteurs	Initialiser compteur
S_CD	Z_RUECK	Compteurs	Paramétrage et compteur décrémental
S_CU	Z_VORW	Compteurs	Paramétrage et compteur incrémental
S_CUD	ZAEHLER	Compteurs	Paramétrage et compteur incrémental/décrémental
SD	SE	Temporisations	Paramétrer et démarrer une temporisation sous forme de retard à la montée
SE	SV	Temporisations	Temporisation sous forme d'impulsion prolongée
SF	SA	Temporisations	Paramétrer et démarrer une temporisation sous forme de retard à la retombée
SHL_DW	SHL_DW	Décalage/rotation	Décalage vers la gauche d'un double mot
SHL_W	SHL_W	Décalage/rotation	Décalage vers la gauche d'un mot
SHR_DI	SHR_DI	Décalage/rotation	Décalage vers la droite d'un entier de 32 bits
SHR_DW	SHR_DW	Décalage/rotation	Décalage vers la droite d'un double mot
SHR_I	SHR_I	Décalage/rotation	Décalage vers la droite d'un entier de 16 bits
SHR_W	SHR_W	Décalage/rotation	Décalage vers la droite d'un mot
SIN	SIN	Fonction sur nombres à virgule flottante	Fonctions trigonométriques d'angles sous forme de nombres réels
S_ODT	S_EVERZ	Temporisations	Paramétrer et démarrer une temporisation sous forme de retard à la montée
S_ODTS	S_SEVERZ	Temporisations	Paramétrer et démarrer une temporisation sous forme de retard à la montée mémorisé
S_OFFDT	S_AVERZ	Temporisations	Paramétrer et démarrer une temporisation sous forme de retard à la retombée
SP	SI	Temporisations	Temporisation sous forme d'impulsion
S_PEXT	S_VIMP	Temporisations	Paramétrer et démarrer une temporisation sous forme d'impulsion prolongée
S_PULSE	S_IMPULS	Temporisations	Paramétrer et démarrer une temporisation sous forme d'impulsion
SQR	SQR	Fonction sur nombres à virgule flottante	Carré d'un nombre réel
SQRT	SQRT	Fonction sur nombres à virgule flottante	Racine carrée d'un nombre réel
SR	SR	Combinaison sur bits	Bascule mise à 1, mise à 0
SS	SS	Temporisations	Temporisation sous forme de retard à la montée mémorisé
SUB_DI	SUB_DI	Fonction sur nombres entiers	Soustraire entiers de 32 bits
SUB_I	SUB_I	Fonction sur nombres entiers	Soustraire entiers de 16 bits
SUB_R	SUB_R	Fonction sur nombres à virgule flottante	Soustraire nombres réels
TAN	TAN	Fonction sur nombres à virgule flottante	Fonctions trigonométriques d'angles sous forme de nombres réels
TRUNC	TRUNC	Conversion	Tronquer à la partie entière (32 bits)
UO	UO	Bits d'état	Bit d'anomalie "Opération illicite"
WAND_DW	WAND_DW	Combinaison sur mots	ET double mot
WAND_W	WAND_W	Combinaison sur mots	ET mot
WOR_DW	WOR_DW	Combinaison sur mots	OU double mot
WOR_W	WOR_W	Combinaison sur mots	OU mot
WXOR_DW	WXOR_DW	Combinaison sur mots	OU exclusif double mot
WXOR_W	WXOR_W	Combinaison sur mots	OU exclusif mot
XOR	XOR	Combinaison sur bits	Combinaison OU exclusif

B Exemples de programmation

B.1 Vue d'ensemble des exemples de programmation

Applications pratiques

Chacune des opérations LOG déclenche une fonction précise. En combinant ces opérations dans un programme, vous pouvez exécuter une grande variété de tâches d'automatisation. Vous trouvez dans la suite quelques exemples d'applications pratiques :

- Commande d'un tapis roulant à l'aide d'opérations de combinaison sur bits
- Détection du sens de déplacement d'un tapis roulant à l'aide d'opérations de combinaison sur bits
- Génération d'une période d'horloge à l'aide d'opérations de temporisation
- Surveillance de l'espace de stockage à l'aide à l'aide d'opérations de comptage et de comparaison
- Calculs à l'aide d'opérations arithmétiques sur nombres entiers
- Réglage de la durée de chauffage d'un four

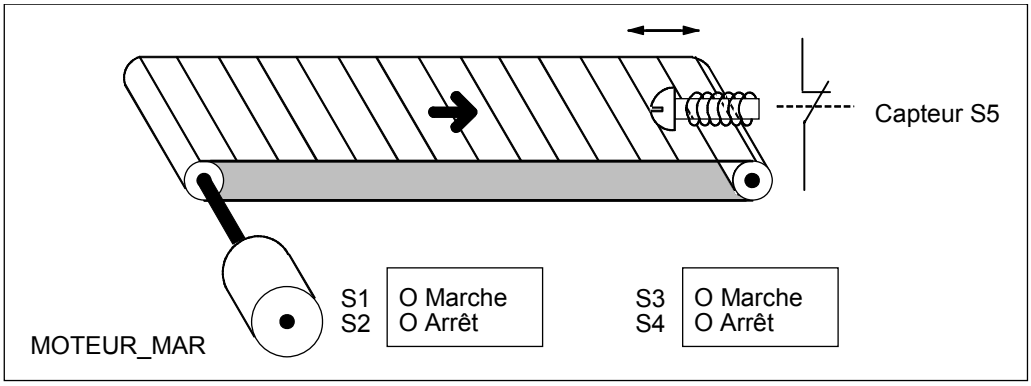
Opérations utilisées

Abréviation Allemande	Catalogue des éléments de programme	Description
WAND_W	Combinaison sur mots	ET mot
WOR_W	Combinaison sur mots	OU mot
Z_RUECK	Compteurs	Décrémenter
Z_VORW	Compteurs	Incrémenter
R	Combinaison sur bits	Mettre à 0
S	Combinaison sur bits	Mettre à 1
P	Combinaison sur bits	Détecter front montant du RLG
ADD_I	Fonction sur nombres entiers	Additionner entiers de 16 bits
DIV_I	Fonction sur nombres entiers	Diviser entiers de 16 bits
MUL_I	Fonction sur nombres entiers	Multiplier entiers de 16 bits
CMP >=I	Comparaison	Comparer entiers de 16 bits
CMP <=I	Comparaison	Comparer entiers de 16 bits
&	Combinaison sur bits	Porte ET
>=1	Combinaison sur bits	Porte OU
=	Combinaison sur bits	Affectation
JMPN	Sauts	Saut si 0
RET	Gestion d'exécution de programme	Retour
MOVE	Transfert	Affecter valeur
SV	Temporisations	Temporisation sous forme d'impulsion prolongée

B.2 Exemples : Opérations combinatoires sur bits

Exemple 1 : Commande d'un tapis roulant

La figure suivante montre un tapis roulant pouvant être mis en route électriquement. Deux boutons-poussoirs, S1 pour MARCHE et S2 pour ARRET, se situent au début du tapis et deux, S3 pour MARCHE et S4 pour ARRET, à la fin du tapis. Il est donc possible de démarrer et d'arrêter le tapis à ses deux extrémités. D'autre part, le capteur S5 arrête le tapis lorsqu'un objet atteint la fin du tapis.



Programmation absolue et symbolique

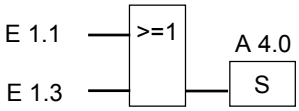
Vous pouvez écrire le programme de commande du tapis roulant en représentant les diverses composants du système convoyeur à l'aide **d'adresses absolues** ou à l'aide de **mnémoniques**.

Vous mettez les mnémoniques choisis dans la table des mnémoniques en relation avec les adresses absolues (voir l'aide en ligne de STEP 7).

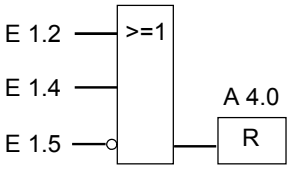
Composant du système	Adresse absolue	Mnémonique	Table de mnémoniques
Bouton-poussoir Marche	E 1.1	S1	E 1.1 S1
Bouton-poussoir Arrêt	E 1.2	S2	E 1.2 S2
Bouton-poussoir Marche	E 1.3	S3	E 1.3 S3
Bouton-poussoir Arrêt	E 1.4	S4	E 1.4 S4
Capteur	E 1.5	S5	E 1.5 S5
Moteur	A 4.0	MOTEUR_MAR	A 4.0 MOTEUR_MAR

Logigramme pour commander un tapis roulant

Réseau 1: Appuyer sur l'un des deux boutons Marche fait démarrer le moteur.

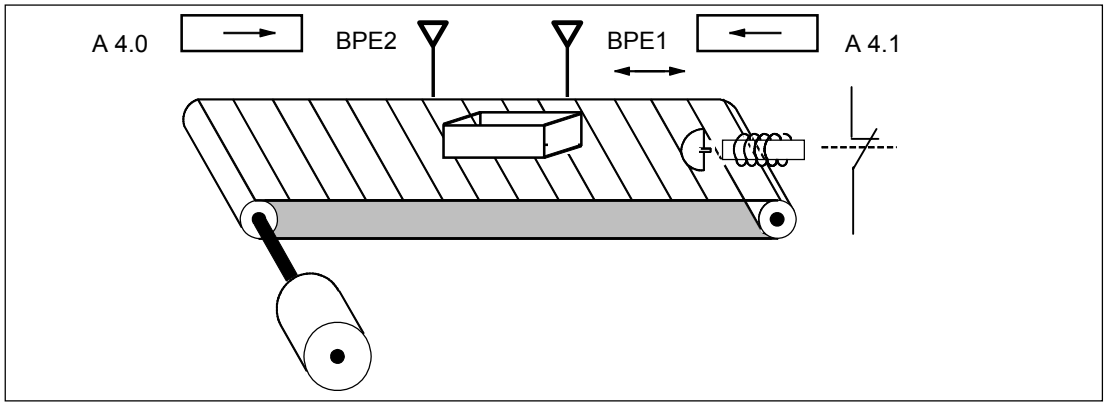


Réseau 2 : Appuyer sur l'un des deux boutons Arrêt ou ouvrir le contact à ouverture à la fin du tapis arrête le moteur.



Exemple 2 : Détection du sens de déplacement d'un tapis roulant

La figure suivante montre un tapis roulant équipé de deux barrières photoélectriques (BPE1 et BPE2) chargées de détecter le sens dans lequel se déplace un paquet sur le tapis. Chaque barrière photoélectrique fonctionne comme un contact à fermeture.



Programmation absolue et symbolique

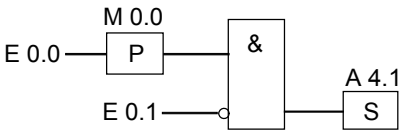
Vous pouvez écrire le programme de commande du tapis roulant en représentant les diverses composants du système convoyeur à l'aide **d'adresses absolues** ou à l'aide de **mnémoniques**.

Vous mettez les mnémoniques choisis dans la table des mnémoniques en relation avec les adresses absolues (voir l'aide en ligne de STEP 7).

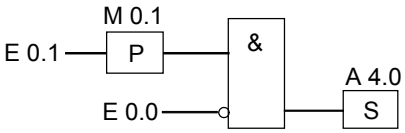
Composant du système	Adresse absolue	Mnémonique	Table de mnémoniques
Barrière photoélectrique 1	E 1.1	BPE1	E 0.0 BPE 1
Barrière photoélectrique 2	E 0.0	BPE2	E 0.1 BPE 2
Affichage pour mouvement vers la droite	A 4.0	DROITE	A 4.0 DROITE
Affichage pour mouvement vers la gauche	A 4.1	GAUCHE	A 4.1 GAUCHE
Mémento de cadence 1	M 0.0	MP1	M 0.0 MP1
Mémento de cadence 2	M 0.1	MP2	M 0.1 MP2

Logigramme pour détecter le sens de déplacement d'un tapis roulant

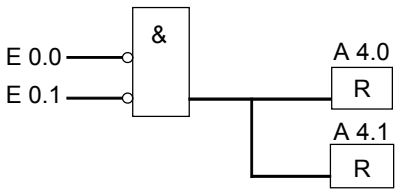
Réseau 1 : Si l'état de signal à l'entrée E 0.0 passe de 0 à 1 (front montant) et si l'état de signal à l'entrée E 0.1 est simultanément à 0, le paquet sur le tapis se déplace vers la gauche.



Réseau 2 : Si l'état de signal à l'entrée E 0.1 passe de 0 à 1 (front montant) et si l'état de signal à l'entrée E 0.0 est simultanément à 0, le paquet sur le tapis se déplace vers la droite. Si l'une des barrières photoélectriques est interrompue, cela signifie qu'un paquet se trouve entre les deux barrières.



Réseau 3: Si une des barrières photoélectriques est interrompue, un paquet se trouve entre les barrières. L'indicateur de sens se désactive.



B.3 Exemple : Opérations de temporisation

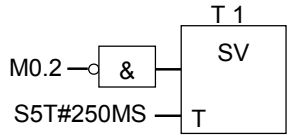
Générateur d'horloge

Vous pouvez utiliser, pour produire un signal qui se répète périodiquement, un générateur d'impulsions d'horloge ou un relais clignotant. On trouve souvent des générateurs d'horloge dans les systèmes de signalisation qui commandent le clignotement des lampes de signalisation.

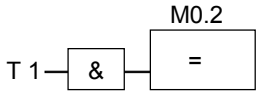
Dans l'automate S7-300, vous pouvez réaliser la génération d'impulsions d'horloge en utilisant le traitement commandé par horloge dans des blocs d'organisation spéciaux. Toutefois, l'exemple présenté dans le programme LOG suivant illustre l'utilisation de fonctions de temporisation pour générer une période d'horloge.

Logigramme pour générer une période d'horloge (rapport d'impulsion 1:1)

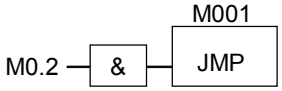
Réseau 1 : Si l'état de signal de la temporisation T1 est 0, charger la valeur de temps 250 ms dans T1 et démarrer T1 sous forme d'impulsion prolongée.



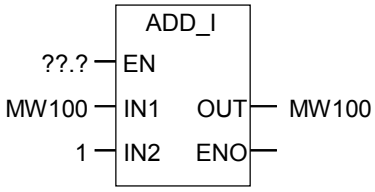
Réseau 2 : L'état de la temporisation est provisoirement mémorisé dans un mémento auxiliaire.



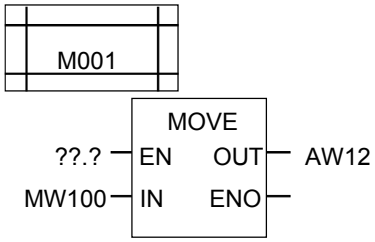
Réseau 3 : Si l'état de signal de la temporisation T1 est 1, sauter au repère de saut M001.



Réseau 4 : Le mot de mémento MW100 est incrémenté de 1 à chaque fois que la temporisation s'est écoulée.

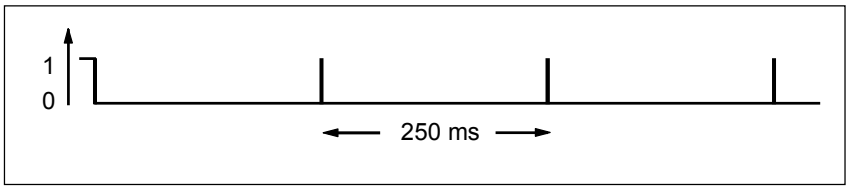


Réseau 5 : L'opération **MOVE** vous permet de voir les différentes fréquences d'horloge aux sorties A 12.0 à A 13.7.



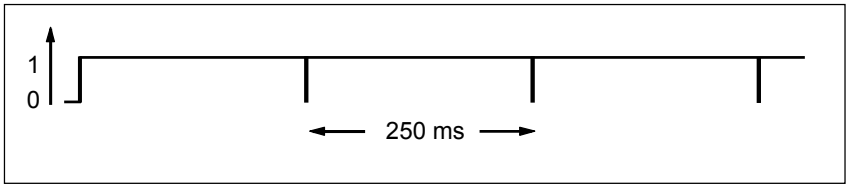
L'interrogation de l'état de signal

L'interrogation du signal de temporisation T1 entraîne le résultat logique suivant pour le paramètre d'entrée inversé de la porte ET (M0.2) dans l'exemple de période d'horloge :



La temporisation est redémarrée une fois le temps écoulé. De ce fait, l'interrogation de l'état de signal ne délivre l'état de signal 1 que brièvement.

La figure montre comment se présente le bit RLG inversé.



Le bit RLG est égal à 0 toutes les 250 ms. Le saut est ignoré et le contenu du mot de mémento MW100 est incrémenté de 1.

Obtenir une fréquence précise

Vous pouvez obtenir les fréquences suivantes avec les bits de l'octet de mémento MB101 et MB100 :

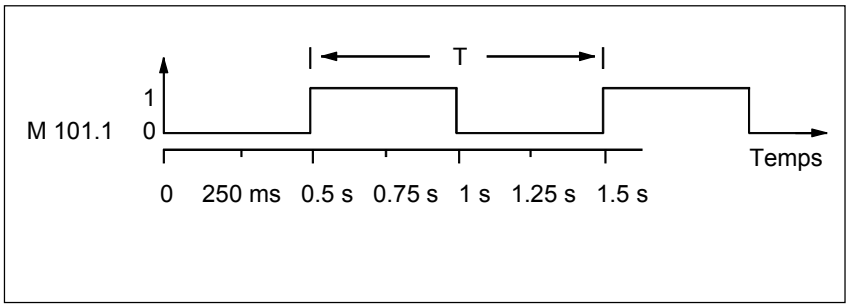
Bits de MB101/ MB100	Fréquence en hertz	Durée	
M 101.0	2.0	0.5 s	(250 ms marche / 250 ms arrêt)
M 101.1	1.0	1 s	(0.5 s marche / 0.5 s arrêt)
M 101.2	0.5	2 s	(1 s marche / 1 s arrêt)
M 101.3	0.25	4 s	(2 s marche / 2 s arrêt)
M 101.4	0.125	8 s	(4 s marche / 4 s arrêt)
M 101.5	0.0625	16 s	(8 s marche / 8 s arrêt)
M 101.6	0.03125	32 s	(16 s marche / 16 s arrêt)
M 101.7	0.015625	64 s	(32 s marche / 32 s arrêt)
M 100.0	0.0078125	128 s	(64 s marche / 64 s arrêt)
M 100.1	0.0039062	256 s	(128 s marche / 128 s arrêt)
M 100.2	0.0019531	512 s	(256 s marche / 256 s arrêt)
M 100.3	0.0009765	1024 s	(512 s marche / 512 s arrêt)
M 100.4	0.0004882	2048 s	(1024 s marche / 1024 s arrêt)
M 100.5	0.0002441	4096 s	(2048 s marche / 2048 s arrêt)
M 100.6	0.000122	8192 s	(4096 s marche / 4096 s arrêt)
M 100.7	0.000061	16384 s	(8192 s marche / 8192 s arrêt)

Etat de signal des bits de l'octet de mémento MB101

Cycle	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Valeur de temps (ms)
0	0	0	0	0	0	0	**0**	0	250
1	0	0	0	0	0	0	**0**	1	250
2	0	0	0	0	0	0	1	0	250
3	0	0	0	0	0	0	1	1	250
4	0	0	0	0	0	1	**0**	0	250
5	0	0	0	0	0	1	**0**	1	250
6	0	0	0	0	0	1	1	0	250
7	0	0	0	0	0	1	1	1	250
8	0	0	0	0	1	0	**0**	0	250
9	0	0	0	0	1	0	**0**	1	250
10	0	0	0	0	1	0	1	0	250
11	0	0	0	0	1	0	1	1	250
12	0	0	0	0	1	1	**0**	0	250

Etat de signal du bit 1 du MB101 (M 101.1)

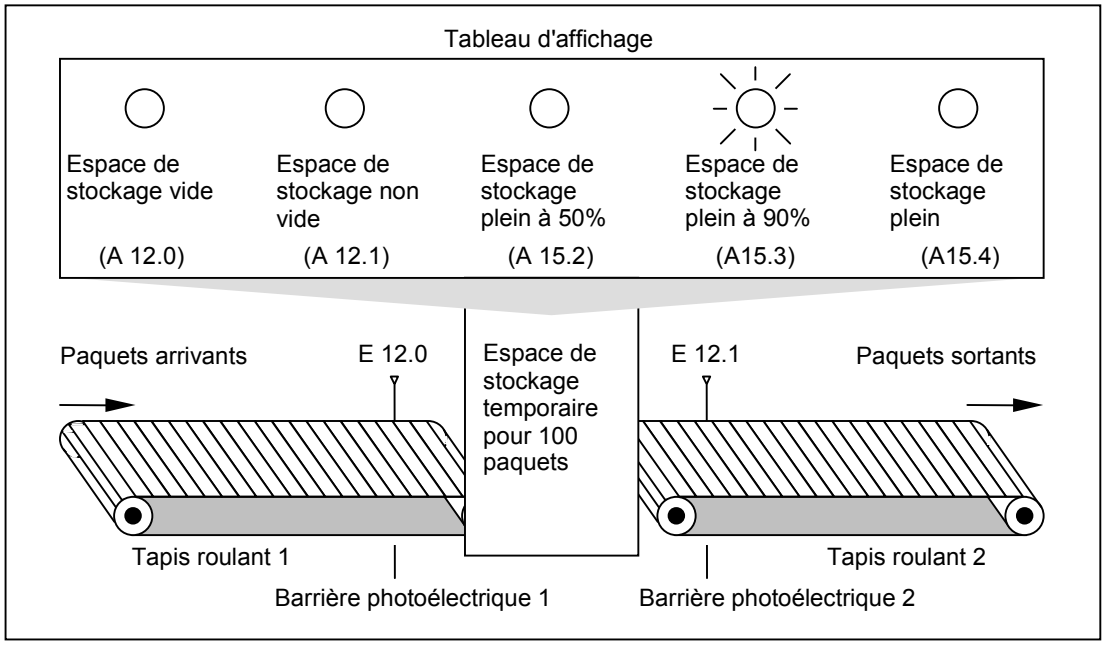
Fréquence = 1/T = 1/1 s = 1 Hz



B.4 Exemple : Opérations de comptage et de comparaison

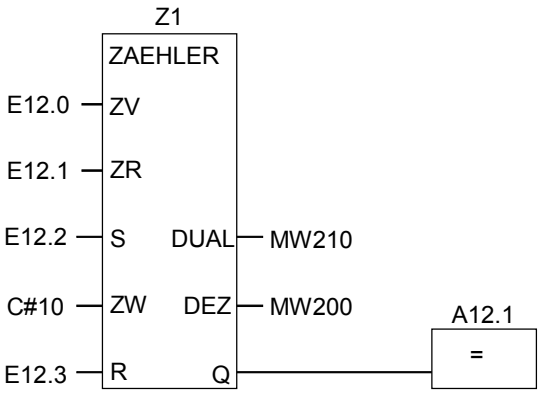
Espace de stockage avec compteur et comparateur

La figure suivante montre un système avec deux tapis roulants et un espace de stockage temporaire entre eux. La tapis roulant 1 transporte les paquets dans l'espace de stockage. Une barrière photoélectrique à l'extrémité du tapis roulant 1, près de l'espace de stockage, détermine le nombre de paquets qui y sont amenés. Le tapis roulant 2 transporte les paquets de l'espace de stockage temporaire à une rampe de chargement d'où ils sont chargés dans des camions afin d'être livrés aux clients. Une barrière photoélectrique à l'extrémité du tapis roulant 2 près de l'espace de stockage détermine le nombre de paquets transportés de l'espace de stockage à la rampe de chargement. Un tableau d'affichage avec cinq lampes indique le niveau de remplissage de l'espace de stockage temporaire.

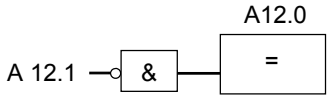


Logigramme pour activer les lampes de signalisation sur un tableau d'affichage

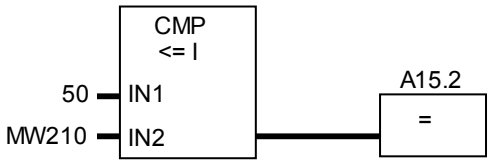
Réseau 1 : En présence d'un front montant à l'entrée ZV, la valeur du compteur Z1 est augmentée de 1 ; en présence d'un front descendant à l'entrée ZR, elle est diminuée de 1. En présence d'un front montant à l'entrée S, la valeur du compteur est mise à la valeur de ZW. En présence d'un front montant à l'entrée R, la valeur du compteur est remise à zéro. La valeur actuelle du compteur Z1 est mémorisée dans le mot de mémento MW200. La lampe de signalisation A 12.1 indique : "Espace de stockage non vide".



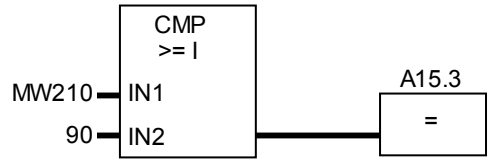
Réseau 2 : La lampe de signalisation A 12.0 indique : "Espace de stockage vide".



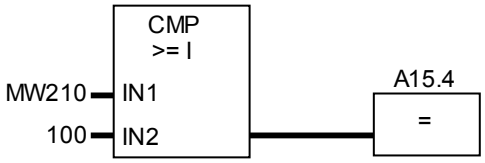
Réseau 3 : Si la valeur 50 est inférieure ou égale à la valeur du compteur (c'est-à-dire que la valeur de comptage est supérieure ou égale à 50), la lampe de signalisation "Espace de stockage plein à 50 %" s'allume.



Réseau 4 : Si la valeur du compteur est supérieure ou égale à 90, la lampe de signalisation "Espace de stockage plein à 90 %" s'allume.



Réseau 5 : Si la valeur du compteur est supérieure ou égale à 100, la lampe de signalisation "Espace de stockage plein" s’allume. Utilisez la sortie A 4.4 pour bloquer le tapis roulant 1.



B.5 Exemple : Opérations arithmétiques sur nombres entiers

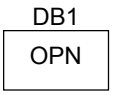
Calcul d'une Équation

L'exemple de programme suivant montre comment obtenir en utilisant trois opérations arithmétiques sur nombres entiers le même résultat que montre l'équation suivante :

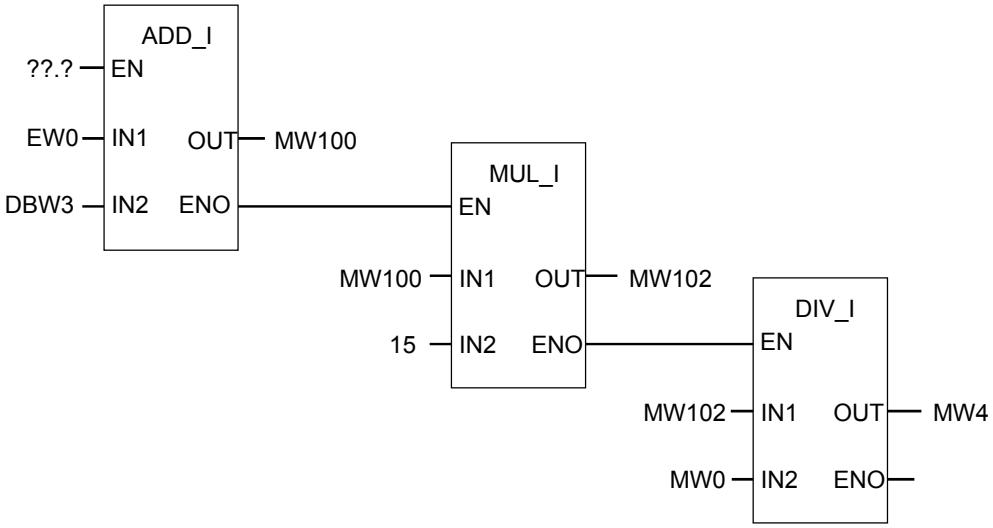
MW4 = ((EW0 + DBW3) x 15) / MW0

Logigramme

Réseau 1 : Ouvrir bloc de données DB1.



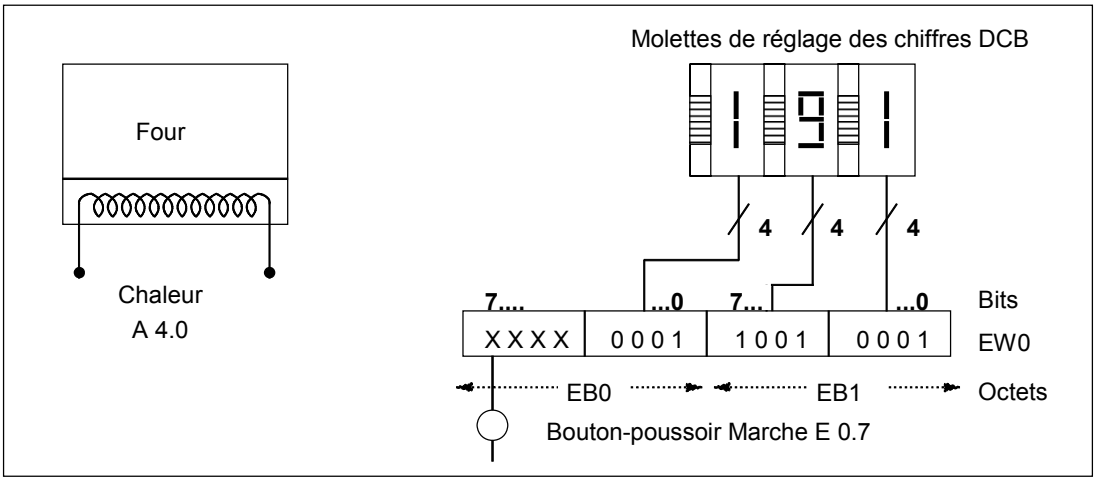
Réseau 2 : Le mot d'entrée EW0 est additionné au mot de données global DBW3 (le bloc de données doit avoir été défini et ouvert) et la somme est chargée dans le mot de mémento MW100. MW100 est ensuite multiplié par 15 et le résultat mémorisé dans le mot de mémento MW102. Puis, MW102 est divisé par MW0 et le résultat mémorisé dans MW4.



B.6 Exemple : Opérations combinatoires sur mots

Chauffage d'un Four

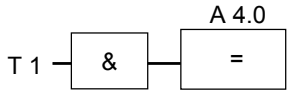
L'opérateur du four déclenche le chauffage du four en appuyant sur le bouton-poussoir Marche. Il peut régler la durée du chauffage à l'aide des molettes représentées dans la figure. La valeur indiquée donne les secondes en format décimal codé binaire (DCB).



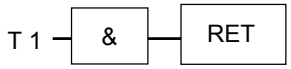
Composants du système	Adresse absolue
Bouton-poussoir Marche	E 0.7
Molette de réglage des unités	E 1.0 à E 1.3
Molette de réglage des dizaines	E 1.4 à E 1.7
Molette de réglage des centaines	E 0.0 à E 0.3
Déclenchement du chauffage	A 4.0

Logigramme pourle chauffage d'un four

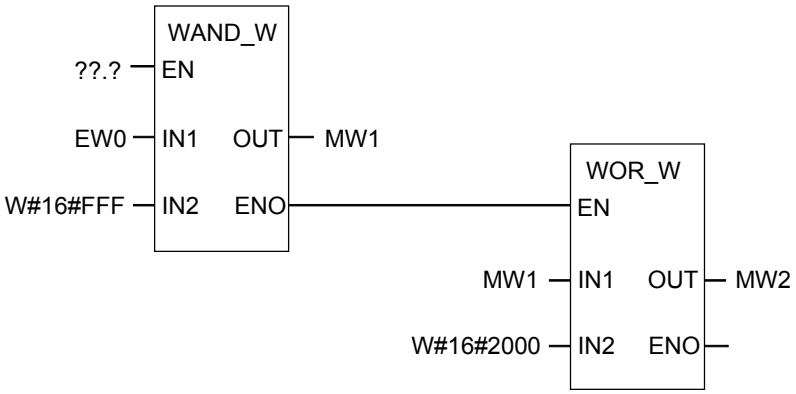
Réseau 1 : Si la temporisation s'exécute, déclencher le chauffage.



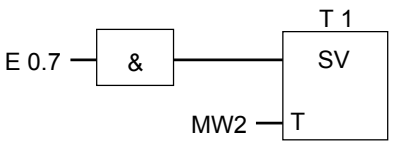
Réseau 2 : Si la temporisation s'exécute, l'opération **Retour** met fin au processus ici.



Réseau 3: Masquer les bits d'entrée E 0.4 à E 0.7 (c'est-à-dire les mettre à 0). Ces bits d'entrée des molettes ne sont pas utilisés. Les 16 bits des entrées correspondant aux molettes sont combinés à W#16#0FFF avec l'opération **ET mot**. Le résultat est chargé dans le mot de mémento MW1. Afin de régler la valeur de temps en secondes, la valeur prédéfinie est combinée à W#16#2000 avec l'opération **OU mot**. Le bit 13 est mis à 1 et le bit 12 est mis à 0.



Réseau 4 : Démarrer la temporisation T1 sous forme d'impulsion prolongée si le bouton-poussoir Marche est enfoncé, en chargeant le mot de mémento MW2 (résultant de la combinaison précédente) comme présélection.



C Pour travailler en LOG

C.1 Mécanisme EN/ENO

L'entrée de validation (EN) et la sortie de validation (ENO) des représentations LOG/CONT sont réalisées à l'aide du bit RB.

Lorsque EN et ENO sont combinées, on a :

ENO = EN AND NOT (erreur de la représentation)

En cas d'absence d'erreur (erreur de la représentation = 0), on a ENO = EN.

Le mécanisme EN/ENO s'utilise pour :

- les opérations arithmétiques,
- les opérations de transfert et de conversion,
- les opérations de décalage et de rotation,
- les appels de blocs.

Ce mécanisme ne s'utilise **pas** pour :

- les comparaisons,
- les compteurs,
- les temporisations.

Pour les instructions effectives de la représentation, des instructions LIST supplémentaires sont générées pour le mécanisme EN/ENO selon la présence de combinaisons précédentes ou suivantes. Les quatre cas possibles sont illustrés par un exemple d'addition.

- Addition avec combinaison de EN et avec combinaison de ENO
- Addition avec combinaison de EN et sans combinaison de ENO
- Addition sans combinaison de EN et avec combinaison de ENO
- Addition sans combinaison de EN et sans combinaison de ENO

Remarques pour la création de vos propres blocs

Lorsque vous souhaitez écrire des blocs que vous voulez appeler dans LOG/CONT, vous devez vous assurer que le bit RB soit mis à 1 lorsque vous quittez le bloc. Le quatrième exemple montre que cela n'est pas automatiquement le cas. Vous ne pouvez pas utiliser le RB comme mémento, car il est continuellement écrasé par le mécanisme EN/ENO. Utilisez à la place une variable temporaire dans laquelle vous enregistrez les erreurs survenues. Initialisez cette variable à 0. A chaque endroit du bloc pour lequel vous pensez qu'une opération erronée est susceptible d'entraîner une erreur pour l'ensemble du bloc, vous mettez cette variable à 1 en vous servant du mécanisme EN/ENO. Il suffit d'utiliser un NOT et une bobine de mise à 1. A la fin du bloc, vous programmez un réseau :

```
fin:   UN erreur
       SAVE
```

Assurez-vous que ce réseau sera parcouru dans tous les cas, ce qui signifie que vous ne devez ni utiliser de BEB dans le bloc, ni sauter ce réseau.

C.1.1 Addition avec combinaison EN et avec combinaison ENO

Si l'addition comporte aussi bien une combinaison EN qu'une combinaison ENO, les instructions LIST suivantes sont initiées :

```
1           U   E   0.0   // Combinaison EN
2           SPBNB _001    // Décaler le RLG dans le RB et sauter si RLG == 0
3           L     in1     // Paramètres de la représentation
4           L     in2     // Paramètres de la représentation
5           +I            // Addition effective
6           T     out     // Paramètres de la représentation
7           UN    OV      // Détection d'erreur
8           SAVE          // Enregistrer l'erreur dans le RB
9           CLR           // Première interrogation
10   _001:  U      RB     // Décaler le RB dans le RLG
11          =     A   4.0
```

Après la première ligne, le RLG contient le résultat de la combinaison précédente. L'instruction SPBNB copie le RLG dans le RB et met le bit de première interrogation à 1.

- Si le RLG est égal à 0, le programme saute à la ligne 10 et poursuit avec U RB. L'addition n'est pas effectuée. Dans la ligne 10, le RB est à nouveau copié dans le RLG et ainsi 0 est affecté à la sortie.
- Si le RLG est égal à 1, le programme ne saute pas plus loin, ce qui signifie que l'addition est effectuée. La ligne 7 permet de déterminer si une erreur s'est produite lors de l'addition, ce qui est enregistré dans le RB à la ligne 8. La ligne 9 met le bit de première interrogation à 1. A la ligne 10, le bit RB est à nouveau copié dans le RLG et ainsi la sortie précise si l'addition a été correctement effectuée.
  Le bit RB n'est plus modifié dans les lignes 10 et 11 et indique donc également si l'addition s'est correctement déroulée.

C.1.2 Addition avec combinaison EN et sans combinaison ENO

Si l'addition comporte une combinaison EN mais pas de combinaison ENO, les instructions LIST suivantes sont initiées :

```
1           U     E      0.0  // Combinaison EN
2           SPBNB _001        // Décaler le RLG dans le RB et sauter si RLG == 0
3           L     in1         // Paramètres de la représentation
4           L     in2         // Paramètres de la représentation
5           +I                // Addition effective
6           T     out         // Paramètres de la représentation
7     _001: NOP   0
```

Après la ligne 1, le RLG contient le résultat de la combinaison précédente. L'instruction SPBNB copie le RLG dans le RB et met le bit de première interrogation à 1.

- Si le RLG est égal à 0, le programme saute à la ligne 7, l'addition n'est pas réalisée, le RLG et le RB valent 0.
- Si le RLG est égal à 1, le programme ne saute pas plus loin, ce qui signifie que l'addition est effectuée. L'éventuelle apparition d'une erreur lors de l'addition n'est pas détectée. Le RLG et le RB valent 1.

C.1.3 Addition sans combinaison EN et avec combinaison ENO

Si l'addition ne comporte pas de combinaison EN mais une combinaison ENO, les instructions LIST suivantes sont initiées :

```
1           L     in1     // Paramètres de la représentation
2           L     in2     // Paramètres de la représentation
3           +I            // Addition effective
4           T     out     // Paramètres de la représentation
5           UN    OV      // Détection d'erreur
6           SAVE          // Enregistrer l'erreur dans le RB
7           CLR           // Première interrogation
8           U     RB      // Décaler le RB dans le RLG
9           =     A      4.0
```

L'addition est réalisée dans tous les cas. La ligne 5 détermine si une erreur s'est produite lors de l'addition, ce qui est enregistré dans le RB à la ligne 6. La ligne 7 met le bit de première interrogation à 1. A la ligne 8, le bit RB est à nouveau copié dans le RLG et ainsi la sortie indique si l'addition s'est correctement déroulée.

Le bit RB n'est plus modifié dans les lignes 8 et 9 et indique donc également si l'addition s'est correctement déroulée.

C.1.4 Addition sans combinaison EN et sans combinaison ENO

Si l'addition ne comporte ni combianison EN, ni combinaison ENO, les instructions LIST suivantes sont initiées :

```
1           L     in1     // Paramètres de la représentation
2           L     in2     // Paramètres de la représentation
3           +I            // Addition effective
4           T     out     // Paramètres de la représentation
5           NOP 0
```

L'addition est effectuée. Le RLG et le bit RB restent inchangés.

C.2 Transmission de paramètres

Les paramètres d'un bloc sont transmis sous forme de valeur. Pour les blocs fonctionnels, une copie de la valeur du paramètre effectif est utilisée dans le DB d'instance au sein du bloc appelé. Pour les fonctions, une copie de la valeur effective se trouve dans la pile des données locales. Les pointeurs ne sont pas copiés. Avant l'appel, les valeurs INPUT sont copiées dans le DB d'instance ou la pile L. Après l'appel, les valeurs OUTPUT sont recopiées dans les variables. Seules des copies sont utilisées au sein du bloc appelé. Les instructions LIST requises se trouvent dans le bloc appelant et restent transparentes à l'utilisateur.

Nota

Si des mémentos, entrées, sorties, périphéries d'entrée ou de sortie sont utilisés en tant qu'opérandes effectifs dans une fonction, ils sont traités de manière différente que les autres opérandes. Leur actualisation n'est effectuée au moyen de la pile L, mais de manière directe.

Exception :

Si le paramètre formel correspondant est un paramètre d'entrée de type de données BOOL, l'actualisation des paramètres effectifs est effectuée via la pile L.

Important

Lors de la programmation du bloc appelé, veillez à compléter les paramètres déclarés comme OUTPUT, sans quoi les valeurs fournies seront aléatoires ! Pour les blocs fonctionnels, on obtiendrait la valeur du DB d'instance inscrite lors du dernier appel, pour les fonctions, la valeur aléatoire se trouvant dans la pile L.

Tenez compte des points suivants :

- Si possible, initialisez tous les paramètres OUTPUT.
- Evitez l'utilisation d'instructions de mise à 1 et de remise à 0, car elles dépendent du RLG. Lorsque le RLG prend la valeur 0, c'est la valeur aléatoire qui est conservée !
- Lorsque vous effectuez un saut au sein du bloc, faites attention de ne pas sauter une ligne dans laquelle sont décrits des paramètres OUTPUT. Tenez également compte de BEB et de l'effet des instructions MCR.

Index

D

E

F

I

J

L

M

N

O

P

R

S

T

U

V

W

X

Z