SIEMENS

SIMATIC

Langage CONT pour SIMATIC S7-300/400

Manuel de référence

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

05/2010
A5E02790081-01

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

A5E02790081-01
℗ 02/2010

Avant-propos

Objet du manuel

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

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

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

CONT correspond au langage « Schéma à contacts » 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.WRI (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 CONT. 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 CONT". 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. Pour les contacts et les bobines, 1 signifie activé ou excité et 0 signifie désactivé ou désexcité.

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 :

- ---| |--- Contact à fermeture
- ---| / |--- Contact à ouverture
- ---(SAVE) Sauvegarder RLG dans RB
- XOR Combinaison OU exclusif
- ---( ) Bobine de sortie
- ---(#)--- Connecteur
- ---|NOT|--- Inverser RLG

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

- Lecture directe en périphérie
- Ecriture directe en périphérie

1.2 ---| |--- Contact à fermeture

Représentation

<opérande>

---| |---

Paramètre	Type de données	Zone de mémoire	Description
<opérande>	BOOL	E, A, M, L, D, T, Z	Bit interrogé

Description de l'opération

---| |--- (Contact à fermeture)

Le contact est fermé si la valeur du bit interrogé sauvegardée en **<opérande>** égale 1. Dans pareil cas, le courant traverse le contact et l'opération fournit un résultat logique (RLG) égal à 1.

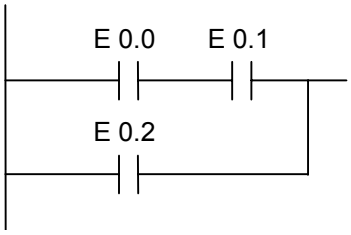
En revanche, si l'état de signal en **<opérande>** est 0, le contact est ouvert : aucun courant ne le traverse et l'opération fournit un résultat logique égal à 0.

S'il s'agit d'une connexion en série, le contact ---| |--- est combiné au RLG bit par bit selon la table de vérité ET. S'il s'agit d'une connexion en parallèle, le contact est combiné au RLG selon la table de vérité OU.

Mot d'état

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

Exemple



Flux d'énergie si l'une des conditions suivantes est satisfaite.

L'état de signal est 1 aux entrées E 0.0 ET E 0.1 OU l'état de signal est 1 à l'entrée E 0.2.

1.3 ---| / |--- Contact à ouverture

Représentation

<opérande>

---| / |---

Paramètre	Type de données	Zone de mémoire	Description
<opérande>	BOOL	E, A, M, L, D, T, Z	Bit interrogé

Description de l'opération

---| / |--- (Contact à ouverture)

Le contact est fermé si la valeur du bit interrogé sauvegardée en **<opérande>** égale 0. Dans ce cas, le courant traverse le contact et l'opération fournit un résultat logique (RLG) égal à 1.

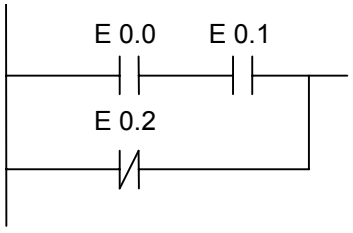
En revanche, si l'état de signal en **<opérande>** est 1, le contact est ouvert : aucun courant ne le traverse et l'opération fournit un résultat logique égal à 0.

S'il s'agit d'une connexion en série, le contact **---| / |---** est combiné au RLG bit par bit selon la table de vérité ET. S'il s'agit d'une connexion en parallèle, le contact est combiné au RLG selon la table de vérité OU.

Mot d'état

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

Exemple



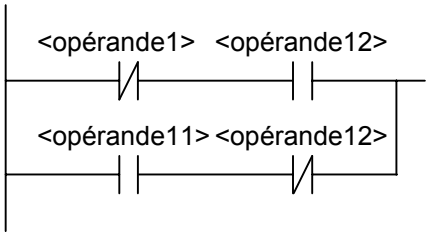
Flux d'énergie si l'une des conditions suivantes est satisfaite :

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.

1.4 XOR Combinaison OU exclusif

Représentation

Cette fonction **XOR** exige un réseau de contacts à ouverture et à fermeture (comme représenté ci-dessous).



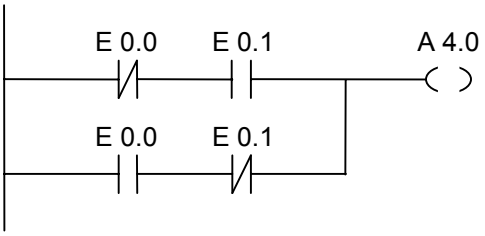
Paramètre	Type de données	Zone de mémoire	Description
<opérande1>	BOOL	E, A, M, L, D, T, Z	Bit interrogé
<opérande2	BOOL	E, A, M, L, D, T, Z	Bit interrogé

Description de l'opération

XOR (Combinaison OU exclusif)

Cette opération génère un RLG égal à 1 si l'état de signal des deux bits précisés est différent.

Exemple



La sortie A 4.0 est mise à 1 si (E 0.0 égale 0 ET E 0.1 égale 1) OU (E 0.0 égale 1 ET E0.1 égale 0).

1.5 ---|NOT|--- Inverser RLG

Représentation

---|NOT|---

Description de l'opération

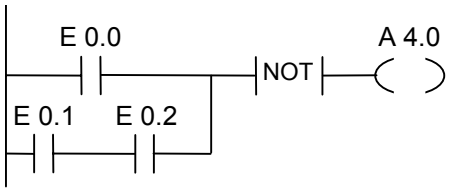
---|NOT|--- (Inverser RLG)

Cette opération inverse le bit de résultat logique (RLG).

Mot d'état

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

Exemple



La sortie A 4.0 est à 0 si l'une des conditions suivantes est satisfaite :

L'état de signal à l'entrée E 0.0 est 1 OU l'état à l'entrée E 0.1. ET à l'entrée E 0.2 est 1.

1.6 ---( ) Bobine de sortie

Représentation

<opérande>

---( )

Paramètre	Type de données	Zone de mémoire	Description
<opérande>	BOOL	E, A, M, L, D	Bit affecté

Description de l'opération

---( ) (Bobine de sortie)

Cette opération fonctionne comme une bobine dans un schéma à relais. Si l'énergie atteint la bobine (RLG = 1), le bit en **<opérande>** est mis à 1. Si l'énergie n'atteint pas la bobine (RLG = 0), le bit en **<opérande>** est mis à 0. Vous ne pouvez placer une sortie qu'à l'extrémité droite d'un trajet de courant. Jusqu'à 16 sorties multiples sont possibles (voir exemples). Vous pouvez créer une sortie inversée à l'aide de l'opération ---|NOT|--- (Inverser RLG).

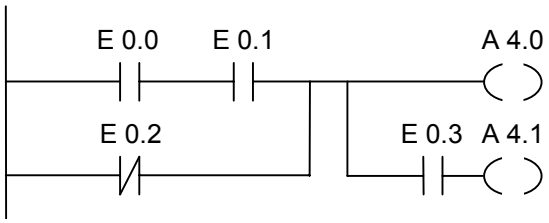
Dépendance par rapport au relais de masquage (Master Control Relay, MCR)

La dépendance par rapport au relais MCR est uniquement activée si une bobine de sortie est dans une zone MCR active. Si le relais MCR est en fonction et que l'énergie atteint une bobine de sortie, le bit adressé prend l'état de signal en cours du flux d'énergie. Si le relais MCR est hors fonction, la valeur 0 est affectée à l'opérande précisé, quel que soit l'état de signal du flux d'énergie.

Mot d'état

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

Exemple



La sortie A 4.0 est à 1 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.

La sortie A 4.1 est à 1 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) ET l'état de signal est 1 à l'entrée E 0.3.

Si le trajet de courant de l'exemple est dans une zone MCR active :

Si le relais MCR est en fonction, l'état de signal des sorties A 4.0 et A 4.1 est fonction de l'état de signal du flux d'énergie comme décrit ci-dessus.

Si le relais MCR est hors fonction, les sorties A 4.0 et A 4.1 sont mises à 0, quel que soit l'état de signal du flux d'énergie.

1.7 ---(#)--- Connecteur

Représentation

<opérande>

---(#)---

Paramètre	Type de données	Zone de mémoire	Description
<opérande>	BOOL	E, A, M, *L, D	Bit affecté

* Un opérande dans la pile des données locales ne peut être utilisé que s'il figure dans la table de déclaration des variables dans la zone TEMP d'un bloc de code (FC, FB, OB).

Description de l'opération

---(#)--- (Connecteur)

Un connecteur est un élément d'affectation intermédiaire qui mémorise le bit RLG (l'état de signal du flux d'énergie) dans l'**<opérande>** précisé. Cet élément sauvegarde la combinaison binaire du dernier branchement ouvert avant lui. S'il s'agit d'une connexion en série avec d'autres éléments, l'opération ---(#)--- est insérée comme un contact. Ne branchez jamais l'élément ---(#)--- à la barre d'alimentation et ne le placez pas immédiatement après un branchement ou comme dernier élément d'une branche. Vous pouvez créer un connecteur inversé ---(#)--- à l'aide de l'opération ---|NOT|--- (Inverser RLG).

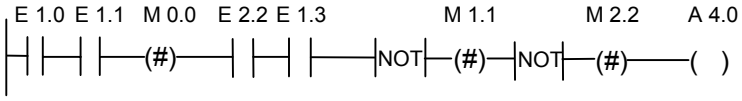
Dépendance par rapport au relais de masquage (Master Control Relay, MCR)

La dépendance par rapport au relais MCR est uniquement activée si un connecteur est dans une zone MCR active. Si le relais MCR est en fonction et que l'énergie atteint un connecteur, le bit adressé prend l'état de signal en cours du flux d'énergie. Si le relais MCR est hors fonction, la valeur 0 est affectée à l'opérande précisé, quel que soit l'état de signal du flux d'énergie.

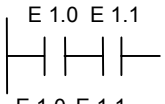
Mot d'état

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

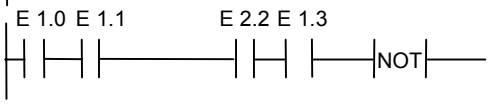
Exemple



M 0.0 a le RLG de



M 1.1 a le RLG de



M 2.2 a le RLG de la combinaison sur bits complète

1.8 ---(R) Mettre à 0

Représentation

<opérande>

---(R)

Paramètre	Type de données	Zone de mémoire	Description
<opérande>	BOOL	E, A, M, L, D, T, Z	Bit mis à 0

Description de l'opération

---(R) (Mettre à 0)

Cette opération ne s'exécute que si le RLG des opérations précédentes a la valeur 1 (flux d'énergie à la bobine). Si l'énergie atteint la bobine (RLG égale 1), l'opération met l'**<opérande>** précisé de l'élément à 0. Si le RLG égale 0 (pas de flux d'énergie à la bobine), l'opération n'a pas d'effet : l'état de signal de l'opérande indiqué de l'élément reste inchangé. Un **<opérande>** peut également être une temporisation (T n°) dont la valeur de temps est mise à 0 ou un compteur (Z n°) dont la valeur de comptage est mise à 0.

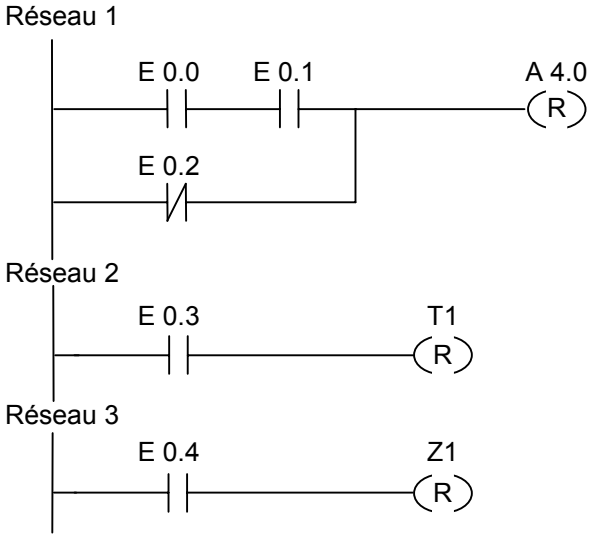
Dépendance par rapport au relais de masquage (Master Control Relay, MCR)

La dépendance par rapport au relais MCR est uniquement activée si une bobine est dans une zone MCR active. Si le relais MCR est en fonction et que l'énergie atteint une bobine, le bit adressé est mis à l'état de signal 0. Si le relais MCR est hors fonction, l'état de signal en cours de l'opérande précisé de l'élément reste inchangé, quel que soit l'état de signal du flux d'énergie.

Mot d'état

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

Exemple



La sortie A 4.0 est uniquement mise à zéro si l'une des conditions suivantes est satisfaite :

(l'état de signal est 1 à l'entrée E 0.0 ET à l'entrée E 0.1) OU l'état de signal est 0 à l'entrée E 0.2.

La temporisation T1 est uniquement mise à zéro si :

l'état de signal à l'entrée E 0.3 égale 1.

Le compteur Z1 est uniquement mis à zéro si :

l'état de signal à l'entrée E 0.4 égale 1.

Si le trajet de courant de l'exemple est dans une zone MCR :

Si le relais MCR est en fonction, la sortie A 4.0, la temporisation T1 et le compteur Z1 sont mis à zéro comme décrit ci-dessus.

Si le relais MCR est hors fonction, la sortie A 4.0, la temporisation T1 et le compteur Z1 restent inchangés, quel que soit l'état de signal du RLG (du flux d'énergie).

1.9 ---(S) Mettre à 1

Représentation

<opérande>

---(S)

Paramètre	Type de données	Zone de mémoire	Description
<opérande>	BOOL	E, A, M, L, D	Bit mis à 1

Description de l'opération

---(S) (Mettre à 1)

Cette opération ne s'exécute que si le RLG des opérations précédentes a la valeur 1 (flux d'énergie à la bobine). Dans ce cas, l'**<opérande>** précisé de l'élément est mis à 1.

Si le RLG égale 0, l'état de signal en cours de l'opérande précisé de l'élément reste inchangé.

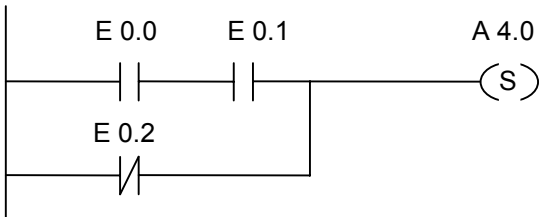
Dépendance par rapport au relais de masquage (Master Control Relay, MCR)

La dépendance par rapport au relais MCR est uniquement activée si une bobine est dans une zone MCR active. Si le relais MCR est en fonction et que l'énergie atteint une bobine, le bit adressé est mis à 1. Si le relais MCR est hors fonction, l'état de signal en cours de l'opérande précisé de l'élément reste inchangé, quel que soit l'état de signal du flux d'énergie.

Mot d'état

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

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) OU l'état de signal est 0 à l'entrée E 0.2.

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

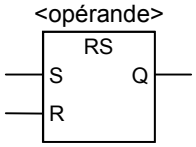
Si le trajet de courant de l'exemple est dans une zone MCR :

Si le relais MCR est en fonction, la sortie A 4.0 est mise à 1 comme décrit ci-dessus.

Si le relais MCR est hors fonction, l'état de signal de la sortie A 4.0 reste inchangé, quel que soit l'état de signal du RLG (du flux d'énergie).

1.10 RS Bascule mise à 0, mise à 1

Représentation



Paramètre	Type de données	Zone de mémoire	Description
<opérande>	BOOL	E, A, M, L, D	Bit mis à 1 ou à 0
S	BOOL	E, A, M, L, D	Valider mise à 1
R	BOOL	E, A, M, L, D	Valider mise à 0
Q	BOOL	E, A, M, L, D	Etat de signal de <opérande>

Description de l'opération

RS (Bascule mise à 0, mise à 1)

Cette opération exécute la mise à 0 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, c'est l'ordre qui compte : la bascule RS exécute d'abord la mise à 0, puis la mise à 1 de l'**<opérande>** indiqué. L'opérande reste donc à 1 pour le reste du cycle du programme.

Les opérations S (mise à 1) et R (mise à 0) s'exécutent uniquement si le RLG égale 1. Si le RLG égale 0, ces opérations ne sont pas influencées et l'opérande indiqué reste inchangé.

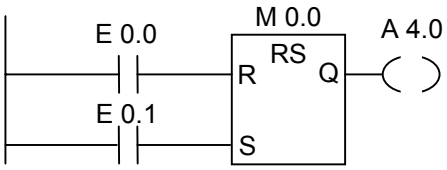
Dépendance par rapport au relais de masquage (Master Control Relay, MCR)

La dépendance par rapport au relais MCR est uniquement activée si l'opération "Bascule mise à 0, mise à 1" est dans une zone MCR active. Si le relais MCR est en fonction, le bit adressé est mis à l'état de signal 1 ou 0 comme décrit ci-dessus. Si le relais MCR est hors fonction, l'état de signal en cours de l'opérande précisé reste inchangé, quel que soit l'état de signal des entrées.

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 bit de 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 bit de 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 les 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.

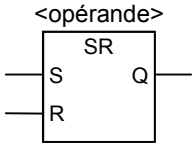
Si l'exemple ci-dessus est dans une zone MCR active :

Si le relais MCR est en fonction, la sortie A 4.0 est mise à 1 ou à 0 comme décrit ci-dessus.

Si le relais MCR est hors fonction, la sortie A 4.0 reste inchangée, quel que soit l'état de signal des entrées.

1.11 SR Bascule mise à 1, mise à 0

Représentation



Paramètre	Type de données	Zone de mémoire	Description
<opérande>	BOOL	E, A, M, L, D	Bit mis à 1 ou à 0
S	BOOL	E, A, M, L, D	Valider mise à 1
R	BOOL	E, A, M, L, D	Valider mise à 0
Q	BOOL	E, A, M, L, D	Etat de signal de <opérande>

Description de l'opération

SR (Bascule mise à 1, mise à 0)

Cette opération exécute la 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, c'est l'ordre qui compte : la bascule SR exécute d'abord la mise à 1, puis la mise à 0 de l'**<opérande>** indiqué. L'opérande reste donc à 0 pour le reste du cycle du programme.

Les opérations S (mise à 1) et R (mise à 0) s'exécutent uniquement si le RLG égale 1. Si le RLG égale 0, ces opérations ne sont pas influencées et l'opérande indiqué reste inchangé.

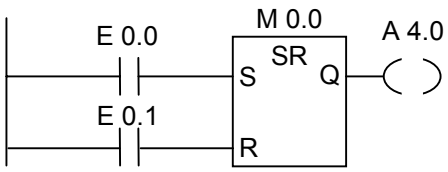
Dépendance par rapport au relais de masquage (Master Control Relay, MCR)

La dépendance par rapport au relais MCR est uniquement activée si l'opération "Bascule mise à 1, mise à 0" est dans une zone MCR active. Si le relais MCR est en fonction, le bit adressé est mis à l'état de signal 1 ou 0 comme décrit ci-dessus. Si le relais MCR est hors fonction, l'état de signal en cours de l'opérande précisé reste inchangé, quel que soit l'état de signal des entrées.

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 bit de 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 bit de 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 les deux la valeur 1, la mise à zéro, exécutée en dernier, l'emporte : M 0.0 est mis à 0 et la sortie A 4.0 est à 0.

Si l'exemple ci-dessus est dans une zone MCR active :

Si le relais MCR est en fonction, la sortie A 4.0 est mise à 1 ou à 0 comme décrit ci-dessus.

Si le relais MCR est hors fonction, la sortie A 4.0 reste inchangée, quel que soit l'état de signal des entrées.

1.12 ---(N)--- Détecter front descendant

Représentation

<opérande>

---(N)---

Paramètre	Type de données	Zone de mémoire	Description
<opérande>	BOOL	E, A, M, L, D	Le mémento de front mémorise l'ancien état de signal du RLG.

Description de l'opération

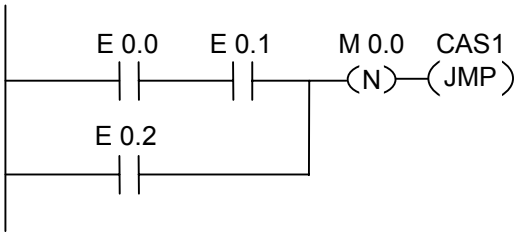
---(N)--- (Détecter front descendant)

Cette opération détecte le passage de 1 à 0 de l'état de signal de l'opérande et montre cette transition avec un RLG égal à 1 après cette opération. L'état de signal en cours du RLG est comparé à l'état de signal de l'opérande (au mémento de front). Si l'état de signal de l'opérande est 1 et le RLG avant l'opération est 0, le résultat logique après l'opération est 1 (impulsion) ; dans tous les autres cas, le résultat logique après l'opération est 0. Le RLG avant l'opération est mémorisé 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 de toute la combinaison binaire. En cas de passage de 1 à 0 du RLG, le programme effectue un saut au repère CAS1.

1.13 ---(P)--- Détecter front montant

Représentation

<opérande>

---(P)---

Paramètre	Type de données	Zone de mémoire	Description
<opérande>	BOOL	E, A, M, L, D	Le mémento de front mémorise l'ancien état de signal du RLG.

Description de l'opération

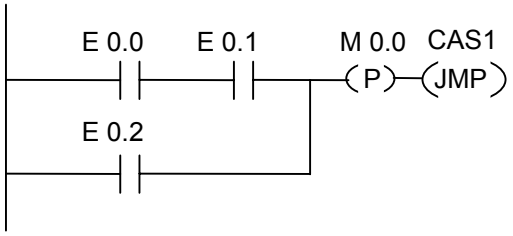
---(P)--- (Détecter front montant du RLG)

Cette opération détecte le passage de 0 à 1 de l'état de signal de l'opérande et montre cette transition avec un RLG égal à 1 après cette opération. L'état de signal en cours du RLG est comparé à l'état de signal de l'opérande (au mémento de front). Si l'état de signal de l'opérande est 0 et le RLG avant l'opération est 1, le résultat logique après l'opération est 1 (impulsion) ; dans tous les autres cas, le résultat logique après l'opération est 0. Le RLG avant l'opération est mémorisé 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 de toute la combinaison binaire. En cas de passage de 0 à 1 du RLG, le programme effectue un saut au repère CAS1.

1.14 ---(SAVE) Sauvegarder RLG dans RB

Représentation

---(SAVE)

Description de l'opération

---(SAVE) (Sauvegarder RLG dans RB)

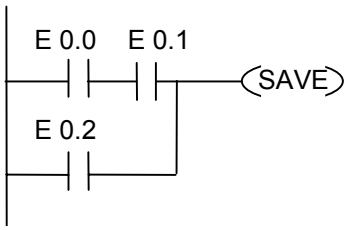
Cette opération sauvegarde le résultat logique (RLG) dans le bit RB (résultat binaire) du mot d'état. Ce faisant, le bit de première interrogation /PI n'est pas mis à zéro.
Pour cette raison, l'état du bit RB est pris en compte en cas de combinaison ET dans le réseau suivant.

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	-	-	-	-	-	-	-	-

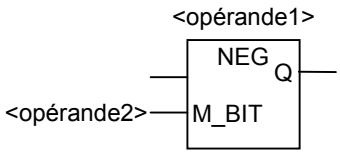
Exemple



L'état de signal du réseau (= RLG) est sauvegardé dans le bit RB.

1.15 NEG Détecter front descendant de signal

Représentation



Paramètre	Type de données	Zone de mémoire	Description
<opérande1>	BOOL	E, A, M, L, D	Signal interrogé
<opérande2>	BOOL	E, A, M, L, D	Le mémento de front M_BIT mémorise l'état de signal précédent de <opérande1>.
Q	BOOL	E, A, M, L, D	Détection de changement d'état de signal

Description de l'opération

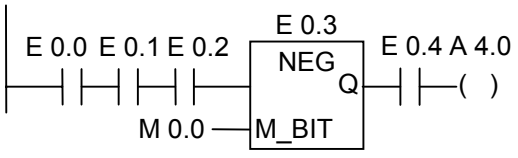
NEG (Détecter front descendant de signal)

Cette opération compare l'état de signal de **<opérande1>** à celui provenant de l'interrogation d'état de signal précédent figurant dans **<opérande2>**. Si l'état de signal en cours du RLG est à 0 et si l'état précédent était 1 (détection d'un front descendant), la sortie Q est à 1 après cette opération ; dans tous les autres cas, elle est à 0.

Mot d'état

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

Exemple

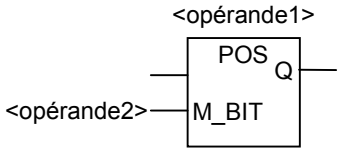


La sortie A 4.0 est à 1 si :

(l'état de signal est 1 aux entrées E 0.0 ET E 0.1 ET E 0.2) ET 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.

1.16 POS Détecter front montant de signal

Représentation



Paramètre	Type de données	Zone de mémoire	Description
<opérande1>	BOOL	E, A, M, L, D	Signal interrogé
<opérande2>	BOOL	E, A, M, L, D	Le mémento de front M_BIT mémorise l'état de signal précédent de <opérande1>.
Q	BOOL	E, A, M, L, D	Détection de changement d'état de signal

Description de l'opération

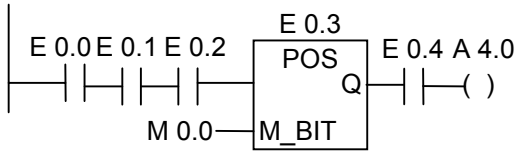
POS (Détecter front montant de signal)

Cette opération compare l'état de signal de **<opérande1>** à celui provenant de l'interrogation d'état de signal précédent figurant dans **<opérande2>**. Si l'état de signal en cours du RLG est à 1 et si l'état précédent était 0 (détection d'un front montant), la sortie Q est à 1 après cette opération ; dans tous les autres cas, elle est à 0.

Mot d'état

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

Exemple



La sortie A 4.0 est à 1 si :

(l'état de signal est 1 aux entrées E 0.0 ET E 0.1 ET E 0.2) ET 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.

1.17 Lecture directe en périphérie

Description de l'opération

La fonction de **lecture directe en périphérie** exige la création d'un réseau (voir l'exemple ci-dessous).

Pour des applications où le temps joue un rôle important, il se peut que l'état de signal en cours d'une entrée TOR doive être lu plus fréquemment que normalement (une fois par cycle). L'opération de lecture directe en périphérie est mise au même état de signal que l'entrée TOR du module d'entrées au moment où le trajet de courant concerné est lu. Sinon, vous devez patienter jusqu'au prochain cycle de l'OB1 lorsque la zone de mémoire des entrées est mise à jour avec l'état de la zone de mémoire de la périphérie.

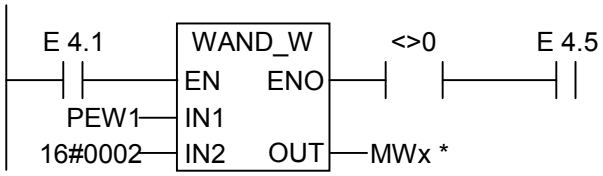
Si vous désirez avoir accès direct en lecture à l'entrée (ou à plusieurs entrées) dans le module d'entrées, utilisez la zone de mémoire de périphérie des entrées (PE) et non la zone de mémoire des entrées (E). La zone de mémoire de la périphérie peut être lue sous forme d'octet, de mot ou de double mot ; une entrée TOR individuelle ne peut donc pas être lue via un contact (bit).

Transmission conditionnelle de la tension en fonction de l'état de signal d'une entrée directe

1. Lecture par la CPU du mot de la zone de mémoire PE contenant les données significatives
2. Ensuite, combinaison par ET du mot de la zone de mémoire PE avec une constante acceptant un résultat différent de 0 si le bit d'entrée est à 1.
3. Contrôle de la condition "différent de 0"

Exemple

Réseau CONT avec l'opération de **lecture directe en périphérie** pour l'entrée E 1.1



* Il faut indiquer MWx pour pouvoir sauvegarder le réseau. x correspond à un numéro autorisé quelconque.

Description de l'opération WAND_W (ET mot) :

PEW1	0000000000101010
W#16#0002	0000000000000010
Résultat	0000000000000010

Dans cet exemple, l'entrée directe E 1.1 est montée en série avec les entrées E 4.1 et E 4.5.

Le mot PEW1 contient l'état de signal direct de E 1.1. PEW1 est combiné à W#16#0002 selon ET. Le résultat est différent de 0 si E 1.1 (deuxième bit) est vrai dans PB1 (1). Le contact U<>0 transfère la tension si le résultat de l'opération WAND_W est différent de zéro.

1.18 Ecriture directe en périphérie

Description de l'opération

La fonction d'**écriture directe en périphérie** exige la création d'un réseau (voir l'exemple ci-dessous).

Pour des applications où le temps joue un rôle important, il se peut que l'état de signal en cours d'une sortie TOR doive être plus fréquemment transféré à un module de sortie que normalement (une fois à la fin d'un cycle de l'OB1). L'opération d'écriture directe en périphérie actualise l'état de signal d'une sortie TOR dans le module de sortie au moment où le trajet de courant concerné est écrit. Sinon, vous devez patienter jusqu'à la fin du cycle de l'OB1 lorsque la zone de mémoire de la périphérie est mise à jour avec l'état de signal de la zone de mémoire des sorties.

Si vous désirez actualiser directement la sortie (ou plusieurs sorties), utilisez la zone de mémoire de périphérie des sorties (PA) et non la zone de mémoire des sorties (A). La zone de mémoire de périphérie des sorties peut être écrite par octet, mot ou double mot ; une sortie TOR individuelle ne peut donc pas être actualisée via une bobine. Pour écrire l'état de signal d'une sortie TOR directement dans un module de sorties, un octet, un mot ou un double mot de la zone de mémoire des sorties A contenant le bit significatif est copié conditionnellement dans la mémoire PA correspondante (dans les opérandes du module de sorties direct).

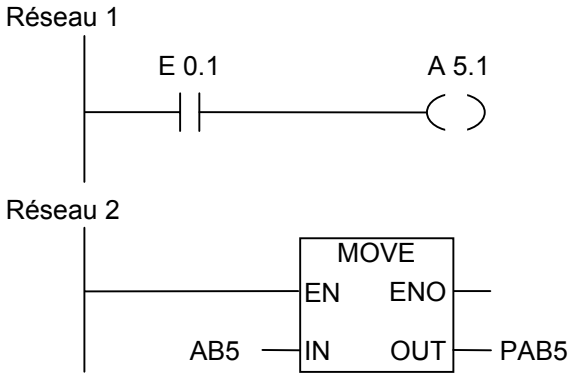
Avertissement

- Comme l'octet complet de la zone de mémoire A est écrit dans le module de sorties, tous les bits de sortie dans l'octet qui est actualisé sont également modifiés si l'opération s'exécute.
- Si un bit de sortie présente un état intermédiaire (1/0) au cours de l'exécution du programme qui ne doit pas être transféré dans le module de sorties, l'opération d'écriture directe en périphérie peut provoquer des états dangereux (impulsions transitoires aux sorties).
- Dans un programme, un module de sorties externe ne doit être adressé qu'une fois comme bobine. En observant cette règle, vous évitez la plupart des problèmes qui risquent d'apparaître en liaison avec l'opération d'écriture directe en périphérie.

Exemple

Réseau CONT avec l'opération d'**écriture directe en périphérie** et le module de sorties TOR 5, voie 1

Les états de signal des bits de l'octet de sortie (AB5) adressé sont soit actualisés, soit ne sont pas modifiés. L'état de signal de E 0.1 est affecté à A 5.1 dans le réseau 1. AB5 est copié dans la zone de mémoire directe correspondante de la périphérie des sorties (PAB5).



Dans cet exemple, la sortie A 5.1 est le bit de sortie exigé.

L'octet PAB5 est mis au même état de signal que le bit de sortie A 5.1.

Les autres bits dans PAB5 sont également actualisés par la copie avec l'opération **MOVE**.

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

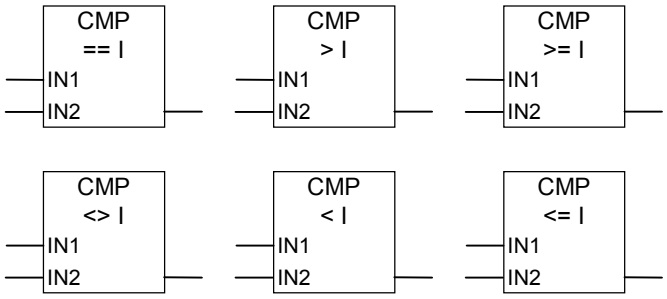
Si la comparaison est vraie, le résultat logique (RLG) est 1. Ce résultat est combiné au RLG du trajet de courant entier selon ET si l'élément de comparaison est utilisé en série ou selon OU s'il est utilisé en parallèle.

Vous disposez des opérations de comparaison suivantes :

- CMP ? I Comparer entiers de 16 bits (16 Bit)
- CMP ? D Comparer entiers de 32 bits (32 Bit)
- CMP ? R Comparer réels

2.2 CMP ? I Comparer entiers de 16 bits

Représentation



Paramètre	Type de données	Zone de mémoire	Description
Entrée de la boîte	BOOL	E, A, M, L, D	Résultat de la combinaison précédente
Sortie de la boîte	BOOL	E, A, M, L, D	Résultat de la comparaison. Utilisé uniquement lorsque le RLG à l'entrée de la boîte est 1.
IN1	INT	E, A, M, L, D ou constante	Premier terme de la comparaison
IN2	INT	E, A, M, L, D ou constante	Second terme de la comparaison

Description de l'opération

CMP ? I (Comparer entiers de 16 bits)

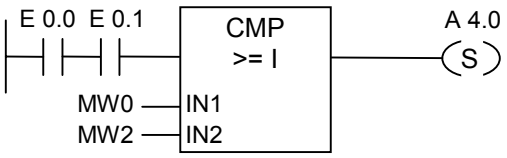
Cette opération que vous pouvez utiliser et placer comme un contact normal compare les entrées IN1 et IN2 selon le type de comparaison que vous avez sélectionné.

Si la comparaison est vraie, le résultat logique (RLG) est 1. Ce résultat est combiné au RLG du trajet de courant entier selon ET si l'élément de comparaison est utilisé en série ou selon OU s'il est utilisé en parallèle.

Mot d'état

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

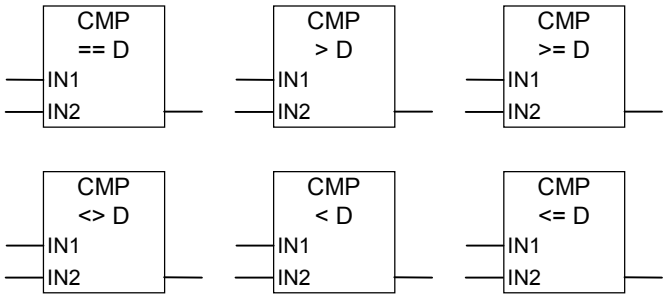
Exemple



La sortie A 4.0 est mise à 1 si l'état de signal est 1 aux entrées E 0.0 ET E 0.1 ET si MW0 >= MW.

2.3 CMP ? D Comparer entiers de 32 bits

Représentation



Paramètre	Type de données	Zone de mémoire	Description
Entrée de la boîte	BOOL	E, A, M, L, D	Résultat de la combinaison précédente
Sortie de la boîte	BOOL	E, A, M, L, D	Résultat de la comparaison. Utilisé uniquement lorsque le RLG à l'entrée de la boîte est 1.
IN1	DINT	E, A, M, L, D ou constante	Premier terme de la comparaison
IN2	DINT	E, A, M, L, D ou constante	Second terme de la comparaison

Description de l'opération

CMP ? D (Comparer entiers de 32 bits)

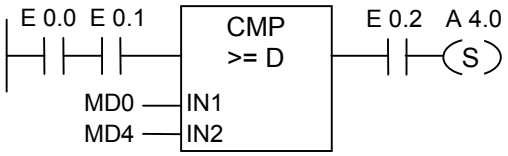
Cette opération que vous pouvez utiliser et placer comme un contact normal compare les entrées IN1 et IN2 selon le type de comparaison que vous avez sélectionné.

Si la comparaison est vraie, son résultat logique (RLG) est 1. Ce résultat est combiné au RLG du trajet de courant selon ET si l'élément de comparaison est utilisé en série ou selon OU s'il est utilisé en parallèle.

Mot d'état

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

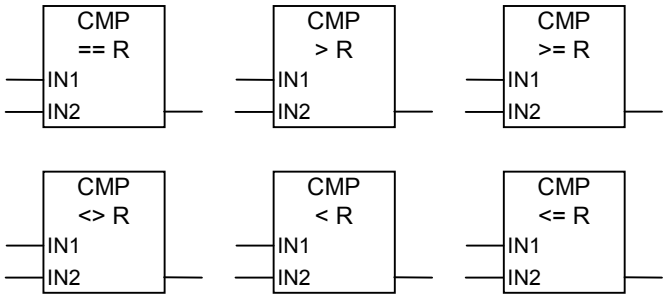
Exemple



La sortie A 4.0 est mise à 1 si l'état de signal est 1 aux entrées E 0.0 ET E 0.1 ET si MD0 >= MD4 ET si l'état de signal est 1 à l'entrée E 0.2.

2.4 CMP ? R Comparer réels

Représentation



Paramètre	Type de données	Zone de mémoire	Description
Entrée de la boîte	BOOL	E, A, M, L, D	Résultat de la combinaison précédente
Sortie de la boîte	BOOL	E, A, M, L, D	Résultat de la comparaison. Utilisé uniquement lorsque le RLG à l'entrée de la boîte est 1.
IN1	REAL	E, A, M, L, D ou constante	Premier terme de la comparaison
IN2	REAL	E, A, M, L, D ou constante	Second terme de la comparaison

Description de l'opération

CMP ? R (Comparer réels)

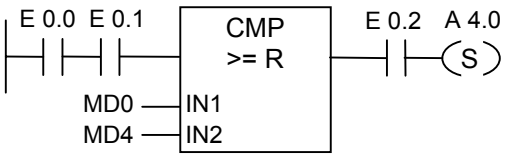
Cette opération que vous pouvez utiliser et placer comme un contact normal compare les entrées IN1 et IN2 selon le type de comparaison que vous avez sélectionné.

Si la comparaison est vraie, le résultat logique (RLG) est 1. Ce résultat est combiné au RLG du trajet de courant entier selon ET si l'élément de comparaison est utilisé en série ou selon OU s'il est utilisé en parallèle.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	x	x	x	x	x	0	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 ET si MD0 >= MD4 ET si l'état de signal est 1 à l'entrée E 0.2.

3 Opérations de conversion

3.1 Vue d'ensemble des opérations de conversion

Description

Les opérations de conversion lisent le contenu du paramètre d'entrée IN, le convertissent ou en changent le signe (par exemple, valeur positive en valeur négative). Le résultat est rangé dans le paramètre de sortie OUT.

Vous disposez des opérations de conversion suivantes :

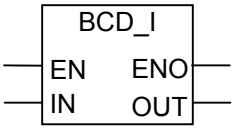
- BCD_I Convertir nombre DCB en entier de 16 bits
- I_BCD Convertir entier de 16 bits en nombre DCB
- BCD_DI Convertir nombre DCB en entier de 32 bits
- I_DI Convertir entier de 16 bits en entier de 32 bits
- DI_BCD Convertir entier de 32 bits en nombre DCB
- DI_R Convertir entier de 32 bits en réel

- 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
- ROUND Arrondir
- TRUNC Tronquer à la partie entière
- CEIL Convertir réel en entier supérieur le plus proche
- FLOOR Convertir 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, L, D	Entrée de validation
ENO	BOOL	E, A, M, L, D	Sortie de validation
IN	WORD	E, A, M, L, D	Nombre en format DCB
OUT	INT	E, A, M, L, D	Valeur entière de 16 bits du nombre DCB

Description de l'opération

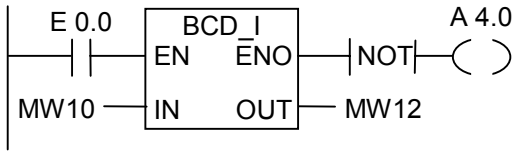
BCD_I (Convertir nombre DCB en entier de 16 bits)

Cette opération 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.

Mot d'état

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

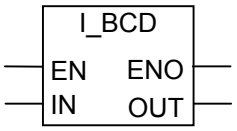
Exemple



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 le mot de mémento MW12. La sortie A 4.0 est mise à 1 si la conversion n'est pas exécutée (ENO = EN = 0).

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, L, D	Entrée de validation
ENO	BOOL	E, A, M, L, D	Sortie de validation
IN	INT	E, A, M, L, D	Nombre entier de 16 bits
OUT	WORD	E, A, M, L, D	Valeur DCB du nombre entier de 16 bits

Description de l'opération

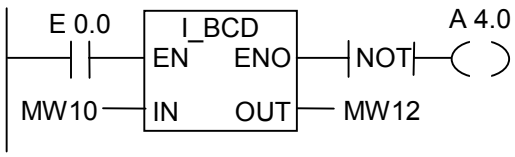
I_BCD (Convertir entier de 16 bits en nombre DCB)

Cette opération lit le contenu du paramètre d'entrée IN comme valeur entière 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

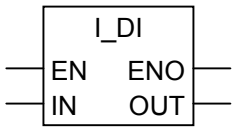
Exemple



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 le mot de mémento MW12. La sortie A 4.0 est mise à 1 en cas de débordement ou si la conversion n'est pas exécutée (E0.0 = 0).

3.4 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, L, D	Entrée de validation
ENO	BOOL	E, A, M, L, D	Sortie de validation
IN	INT	E, A, M, L, D	Valeur entière de 16 bits à convertir
OUT	DINT	E, A, M, L, D	Résultat : nombre entier de 32 bits

Description de l'opération

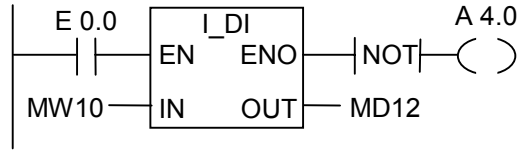
I_DI (Convertir entier de 16 bits en entier de 32 bits)

Cette opération lit le contenu du paramètre d'entrée IN comme valeur entière 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

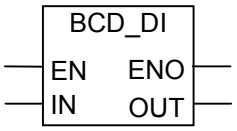
Exemple



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 le double mot de mémento MD12. La sortie A 4.0 est mise à 1 si la conversion n'est pas exécutée (ENO = EN = 0).

3.5 BCD_DI Convertir nombre DCB en entier de 32 bits

Représentation



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
IN	DWORD	E, A, M, L, D	Nombre en format DCB
OUT	DINT	E, A, M, L, D	Valeur entière de 32 bits du nombre DCB

Description de l'opération

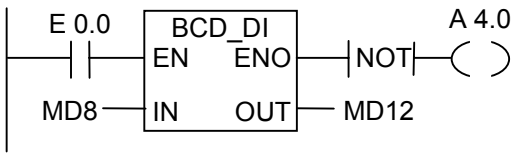
BCD_DI (Convertir nombre DCB en entier de 32 bits)

Cette opération lit le contenu du paramètre d'entrée IN comme nombre décimal codé binaire à sept chiffres (DCB, +/- 9999999), 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

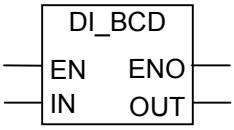
Exemple



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 le double mot de mémento MD12. La sortie A 4.0 est mise à 1 si la conversion n'est pas exécutée (ENO = EN = 0).

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, L, D	Entrée de validation
ENO	BOOL	E, A, M, L, D	Sortie de validation
IN	DINT	E, A, M, L, D	Nombre entier de 32 bits
OUT	DWORD	E, A, M, L, D	Valeur DCB du nombre entier de 32 bits

Description de l'opération

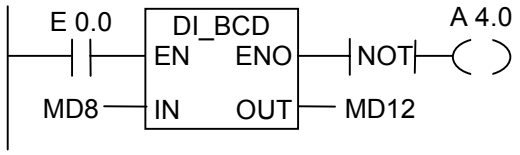
DI_BCD (Convertir entier de 32 bits en nombre DCB)

Cette opération lit le contenu du paramètre d'entrée IN comme valeur entière de 32 bits, le convertit en un nombre décimal codé binaire à sept chiffres (DCB, +/- 9999999) 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

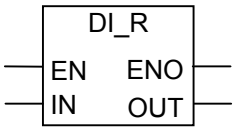
Exemple



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 le double mot de mémento MD12. La sortie A 4.0 est mise à 1 en cas de débordement ou si la conversion n'est pas exécutée (E0.0 = 0).

3.7 DI_R Convertir entier de 32 bits en réel

Représentation



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
IN	DINT	E, A, M, L, D	Nombre entier de 32 bits
OUT	REAL	E, A, M, L, D	Nombre réel

Description de l'opération

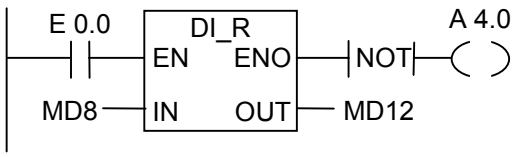
DI_R (Convertir entier de 32 bits en réel)

Cette opération lit le contenu du paramètre d'entrée IN comme nombre entier de 32 bits et le convertit en nombre à virgule flottante. 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

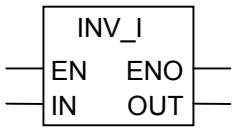
Exemple



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 à virgule flottante. Le résultat est rangé dans le double mot de mémento MD12. La sortie A 4.0 est mise à 1 si la conversion n'est pas exécutée (ENO = EN = 0).

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, L, D	Entrée de validation
ENO	BOOL	E, A, M, L, D	Sortie de validation
IN	INT	E, A, M, L, D	Valeur d'entrée entière de 16 bits
OUT	INT	E, A, M, L, D	Complément à 1 de l'entier de 16 bits dans IN

Description de l'opération

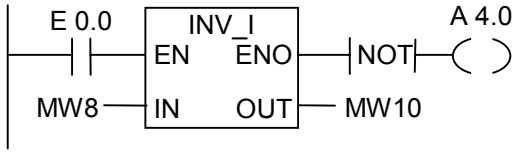
INV_I (Complément à 1 d'entier de 16 bits)

Cette opération lit le contenu du paramètre d'entrée IN et exécute l'opération de combinaison **OU exclusif** avec le masque hexadécimal W#16#FFFF afin d'inverser la valeur de chaque bit. 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

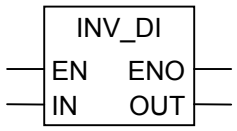


Si l'état de signal est 1 à l'entrée E 0.0, l'état de chaque bit du mot de mémento MW8 est inversé.

MW8 = 01000001 10000001 est converti en MW10 = 10111110 01111110. La sortie A 4.0 est mise à 1 si la conversion n'est pas exécutée (ENO = EN = 0).

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, L, D	Entrée de validation
ENO	BOOL	E, A, M, L, D	Sortie de validation
IN	DINT	E, A, M, L, D	Valeur d'entrée entière de 32 bits
OUT	DINT	E, A, M, L, D	Complément à 1 de l'entier de 32 bits dans IN

Description de l'opération

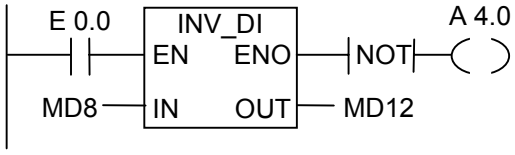
INV_DI (Complément à 1 d'entier de 32 bits)

Cette opération lit le contenu du paramètre d'entrée IN et exécute l'opération de combinaison **OU exclusif** avec le masque hexadécimal W#16#FFFF FFFF afin d'inverser la valeur de chaque bit. 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

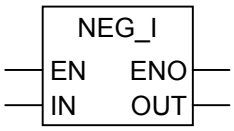


Si l'état de signal est 1 à l'entrée E 0.0, l'état de chaque bit du double mot de mémento MD8 est inversé.

MD8 = F0FF FFF0 est converti en MD12 = 0F00 000F. La sortie A 4.0 est mise à 1 si la conversion n'est pas exécutée (ENO = EN = 0).

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, L, D	Entrée de validation
ENO	BOOL	E, A, M, L, D	Sortie de validation
IN	INT	E, A, M, L, D	Valeur d'entrée entière de 16 bits
OUT	INT	E, A, M, L, D	Complément à 2 de l'entier de 16 bits dans IN

Description de l'opération

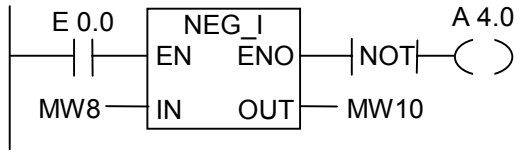
NEG_I (Complément à 2 d'entier de 16 bits)

Cette opération lit le contenu du paramètre d'entrée IN et en change le signe (par exemple, valeur positive en valeur négative). ENO et EN ont toujours un état de signal identique, à l'exception suivante près : si l'état de signal de EN est égal à 1 et qu'il y a débordement, alors 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



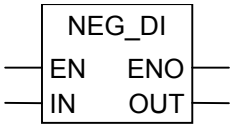
Si l'état de signal est 1 à l'entrée E 0.0, le contenu du mot de mémento MW8 est transmis, avec le signe opposé, du paramètre OUT au mot de mémento MW10.

MW8 = + 10 donne MW10 = - 10. La sortie A 4.0 est mise à 1 si la conversion n'est pas exécutée (ENO = EN = 0).

Si l'état de signal de EN est égal à 1 et qu'il y a débordement, alors l'état de signal de ENO est égal à 0.

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, L, D	Entrée de validation
ENO	BOOL	E, A, M, L, D	Sortie de validation
IN	DINT	E, A, M, L, D	Valeur d'entrée entière de 32 bits
OUT	DINT	E, A, M, L, D	Complément à 2 de l'entier de 32 bits dans IN

Description de l'opération

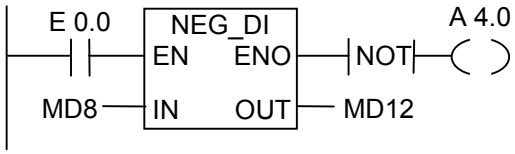
NEG_DI (Complément à 2 d'entier de 32 bits)

Cette opération lit le contenu du paramètre d'entrée IN et en change le signe (par exemple, valeur positive en valeur négative). ENO et EN ont toujours un état de signal identique, à l'exception suivante près : si l'état de signal de EN est égal à 1 et qu'il y a débordement, alors 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



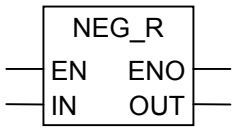
Si l'état de signal est 1 à l'entrée E 0.0, le contenu du double mot de mémento MD8 est transmis, avec le signe opposé, du paramètre OUT au double mot de mémento MD12.

MD8 = + 1000 donne MD12 = - 1000. La sortie A 4.0 est mise à 1 si la conversion n'est pas exécutée (ENO = EN = 0).

Si l'état de signal de EN est égal à 1 et qu'il y a débordement, alors l'état de signal de ENO est égal à 0.

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, L, D	Entrée de validation
ENO	BOOL	E, A, M, L, D	Sortie de validation
IN	REAL	E, A, M, L, D	Valeur d'entrée : nombre réel
OUT	REAL	E, A, M, L, D	Nombre réel du paramètre IN avec inversion de signe

Description de l'opération

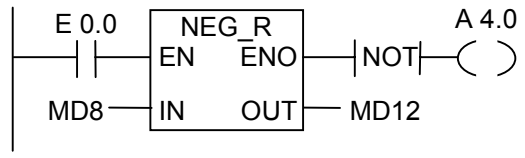
NEG_R (Inverser le signe d'un nombre réel)

Cette opération lit le contenu du paramètre d'entrée IN et en change le signe (par exemple, valeur positive en valeur négative). Elle correspond à une opération de multiplication par (-1). ENO et EN ont toujours un état de signal identique.

Mot d'état

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

Exemple

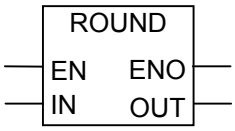


Si l'état de signal est 1 à l'entrée E 0.0, le contenu du double mot de mémento MD8 est transmis, avec le signe inverse, du paramètre OUT au double mot de mémento MD12.

MD8 = + 6,234 donne MD12 = - 6,234. La sortie A 4.0 est mise à 1 si la conversion n'est pas exécutée (ENO = EN = 0).

3.13 ROUND Arrondir

Représentation



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
IN	REAL	E, A, M, L, D	Valeur à arrondir
OUT	DINT	E, A, M, L, D	IN arrondi au nombre entier le plus proche

Description de l'opération

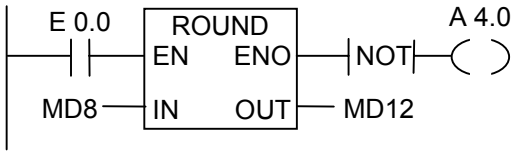
ROUND (Arrondir par excès ou par défaut)

Cette opération lit le contenu du paramètre d'entrée IN comme nombre à virgule flottante 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 le nombre à virgule flottante se situe exactement entre deux nombres entiers, le nombre pair est pris comme résultat. 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

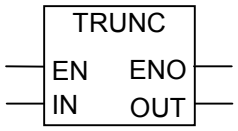
Exemple



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 à virgule flottante et converti en nombre entier de 32 bits. Le résultat de cette fonction "Arrondir par excès ou par défaut" est rangé dans le double mot de mémento MD12. La sortie A 4.0 est mise à 1 en cas de débordement ou si la conversion n'est pas exécutée (E 0.0 = 0).

3.14 TRUNC Tronquer à la partie entière

Représentation



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
IN	REAL	E, A, M, L, D	Nombre à virgule flottante à tronquer
OUT	DINT	E, A, M, L, D	Partie entière de la valeur de IN

Description de l'opération

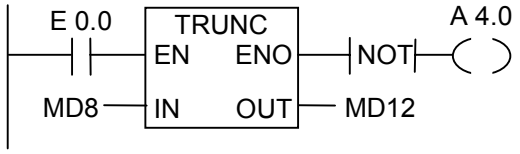
TRUNC (Tronquer à la partie entière)

Cette opération lit le contenu du paramètre d'entrée IN comme nombre réel et le convertit en nombre entier de 32 bits. Le résultat, qui est la partie entière du nombre réel spécifié, est rangé 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

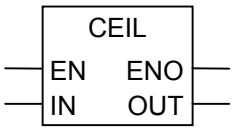
Exemple



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. Le résultat, qui est le composant entier du nombre réel, est rangé dans le double mot de mémento MD12. La sortie A 4.0 est mise à 1 en cas de débordement ou si la conversion n'est pas exécutée (E 0.0 = 0).

3.15 CEIL Convertir réel en entier supérieur le plus proche

Représentation



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
IN	REAL	E, A, M, L, D	Nombre réel à convertir
OUT	DINT	E, A, M, L, D	Nombre entier de 32 bits supérieur le plus proche du nombre réel

Description de l'opération

CEIL (Convertir réel en entier supérieur le plus proche)

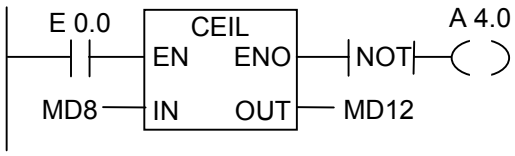
Cette opération lit le contenu du paramètre d'entrée IN comme nombre à virgule flottante et le convertit en un nombre entier de 32 bits. Le résultat est l'entier supérieur le plus proche du nombre réel indiqué (arrondi au nombre entier supérieur le plus proche). En cas de débordement, ENO est mis à 0.

Mot d'état

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

* Fonction exécutée (EN = 1)
** Fonction non exécutée (EN = 0)

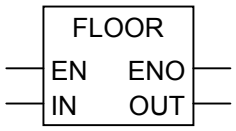
Exemple



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 à virgule flottante et converti en nombre entier de 32 bits selon le principe d'arrondi au nombre entier supérieur ou égal le plus proche. Le résultat est rangé dans le double mot de mémento MD12. La sortie A 4.0 est mise à 1 en cas de débordement ou si la conversion n'est pas exécutée (E 0.0 = 0).

3.16 FLOOR Convertir réel en entier inférieur le plus proche

Représentation



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
IN	REAL	E, A, M, L, D	Nombre réel à convertir
OUT	DINT	E, A, M, L, D	Nombre entier de 32 bits inférieur le plus proche du nombre réel

Description de l'opération

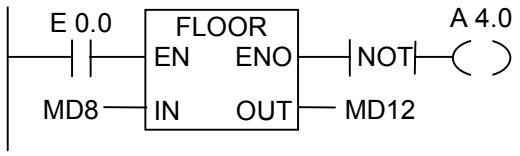
FLOOR (Convertir réel en entier inférieur le plus proche)

Cette opération lit le contenu du paramètre d'entrée IN comme nombre à virgule flottante et le convertit en un nombre entier de 32 bits. Le résultat est l'entier inférieur le plus proche du nombre réel indiqué (arrondi au nombre entier inférieur le plus proche). En cas de débordement, ENO est mis à 0.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture									

Exemple



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 à virgule flottante et converti en nombre entier de 32 bits selon le principe d'arrondi au nombre entier inférieur ou égal le plus proche. Le résultat est rangé dans le double mot de mémento MD12. La sortie A 4.0 est mise à 1 en cas de débordement ou si la conversion n'est pas exécutée (E 0.0 = 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 CONT 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 :

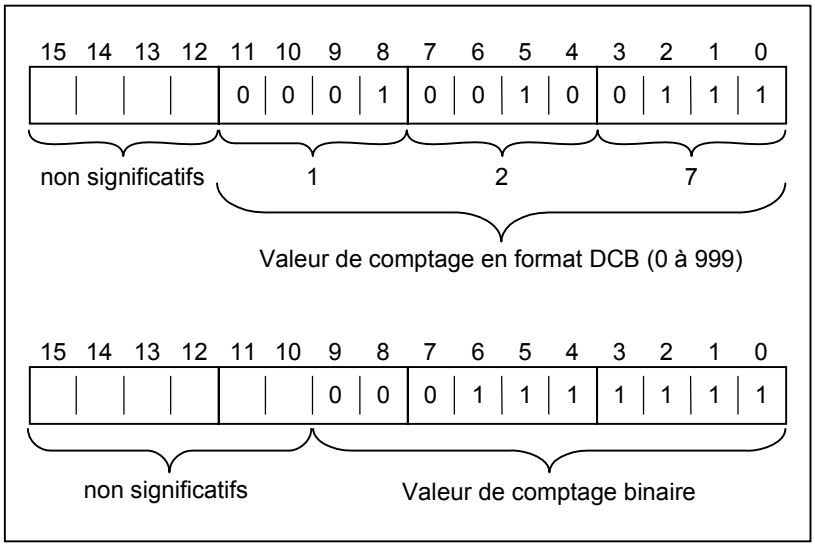
- ZÄHLER Paramétrage et compteur d'incrémentation/décrémentation
- Z_VORW Paramétrage et compteur d'incrémentation
- Z_RUECK Paramétrage et compteur de décrémentation
- ---(SZ) Initialiser compteur
- ---(ZV) Décrémenter
- ---(ZR) Incrémenter

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.

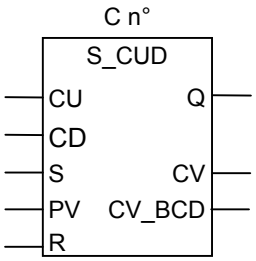
La figure suivante montre le contenu du compteur après le chargement de la valeur 127, ainsi que le contenu de la cellule de compteur après assignation d'une valeur.



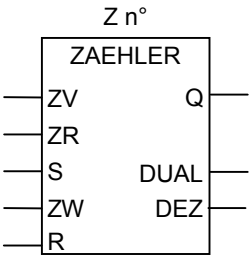
4.2 ZAEHLER Paramétrage et compteur d'incrémentation/décrémentation

Représentation

Anglaise



Allemande



Paramètre Anglaise	Paramètre Allemande	Type de données	Zone de mémoire	Description
C n°	Z n°	COUNTER	Z	Numéro d'identification du compteur. La plage dépend de la CPU.
CU	ZV	BOOL	E, A, M, L, D	Entrée d'incrémentation
CD	ZR	BOOL	E, A, M, L, D	Entrée de décrémentation
S	S	BOOL	E, A, M, L, D	Entrée d'initialisation du compteur
PV	ZW	WORD	E, A, M, L, D ou constante	Valeur de comptage entrée sous forme C#<valeur> dans la plage comprise entre 0 et 999
PV	ZW	WORD	E, A, M, L, D	Valeur d'initialisation du compteur
R	R	BOOL	E, A, M, L, D	Entrée de remise à zéro
CV	DUAL	WORD	E, A, M, L, D	Valeur de comptage en cours (format hexadécimal)
CV_BCD	DEZ	WORD	E, A, M, L, D	Valeur de comptage en cours (format DCB)
Q	Q	BOOL	E, A, M, L, D	Etat du compteur

Description de l'opération

ZAEHLER (Paramétrage et compteur d'incrémentation/décrémentation)

Un front montant à l'entrée S de cette opération initialise le compteur à la valeur figurant dans l'entrée ZW. Un 1 à l'entrée R remet le compteur, et donc la valeur de comptage, à zéro.

Le compteur est incrémenté d'une unité si l'état de signal à l'entrée ZV passe de 0 à 1 – front montant – et que la valeur du compteur est inférieure à 999.

Le compteur est décrémenté d'une unité si l'état de signal à l'entrée ZR passe de 0 à 1 – front montant – et que la valeur du compteur est supérieure à 0.

En cas de front montant aux deux entrées de comptage, les deux fonctions sont exécutées 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.

L'état du signal à la sortie Q est à 1 lorsque la valeur de comptage est supérieure à 0 ; il est à 0 lorsque la valeur de comptage est égale à 0.

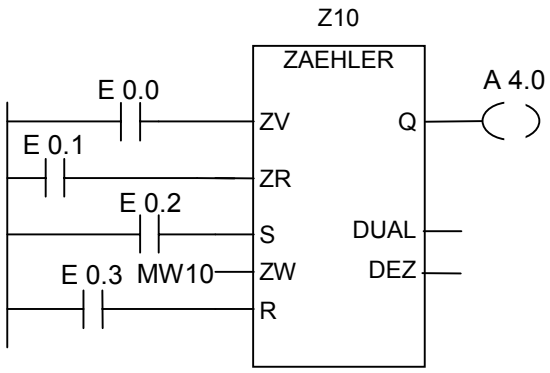
Mot d'état

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

Nota

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

Exemple

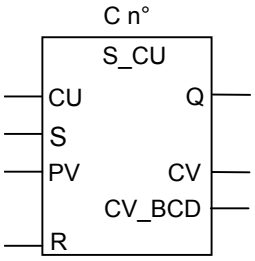


Si l'état de signal à l'entrée E 0.2 passe de 0 à 1, le compteur est initialisé à la valeur figurant dans le mot de mémento MW10. Si l'état de signal en E 0.0 passe de 0 à 1, la valeur du compteur Z10 est incrémentée d'un à moins qu'elle ne soit déjà égale à 999. Si l'état de signal en E 0.1 passe de 0 à 1, la valeur du compteur Z10 est décrémentée d'un à moins qu'elle ne soit déjà égale à 0. L'état de signal de la sortie A 4.0 est 1 si Z10 est différent de zéro.

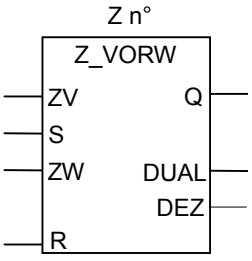
4.3 Z_VORW Paramétrage et compteur d'incrémentation

Représentation

Anglaise



Allemande



Paramètre Anglaise	Paramètre Allemande	Type de données	Zone de mémoire	Description
C n°	Z n°	COUNTER	Z	Numéro d'identification du compteur. La plage dépend de la CPU.
CU	ZV	BOOL	E, A, M, L, D	Entrée d'incrémentation
S	S	BOOL	E, A, M, L, D	Entrée d'initialisation du compteur
PV	ZW	WORD	E, A, M, L, D ou constante	Valeur de comptage entrée sous forme C#<valeur> dans la plage comprise entre 0 et 999
PV	ZW	WORD	E, A, M, L, D	Valeur d'initialisation du compteur
R	R	BOOL	E, A, M, L, D	Entrée de remise à zéro
CV	DUAL	WORD	E, A, M, L, D	Valeur de comptage en cours (format hexadécimal)
CV_BCD	DEZ	WORD	E, A, M, L, D	Valeur de comptage en cours (format DCB)
Q	Q	BOOL	E, A, M, L, D	Etat du compteur

Description de l'opération

Z_VORW (Paramétrage et compteur d'incrémentation)

Un front montant à l'entrée S de cette opération initialise le compteur à la valeur figurant dans l'entrée ZW.

Un 1 à l'entrée R remet le compteur, et donc la valeur de comptage, à zéro.

Le compteur est incrémenté d'une unité si l'état de signal à l'entrée ZV passe de 0 à 1 - front montant - et que la valeur du compteur 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.

L'état de signal de la sortie Q est à 1 lorsque la valeur de comptage est supérieure à 0 ; il est à 0 lorsque la valeur de comptage est égale à 0.

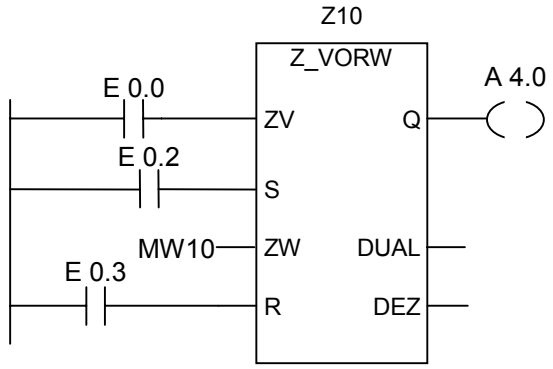
Mot d'état

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

Nota

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

Exemple

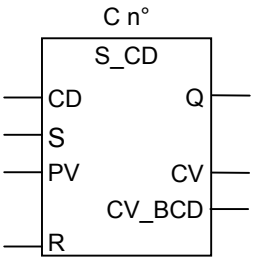


Si l'état de signal à l'entrée E 0.2 passe de 0 à 1, le compteur est initialisé à la valeur figurant dans le mot de mémento MW10. Si l'état de signal en E 0.0 passe de 0 à 1, la valeur du compteur Z10 est incrémentée d'un à moins qu'elle ne soit déjà égale à 999. L'état de signal de la sortie A 4.0 est 1 si Z10 est différent de zéro.

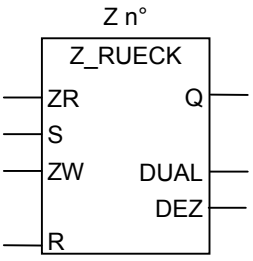
4.4 Z_RUECK Paramétrage et compteur de décrémentation

Représentation

Anglaise



Allemande



Paramètre Anglaise	Paramètre Allemande	Type de données	Zone de mémoire	Description
C n°	Z n°	COUNTER	Z	Numéro d'identification du compteur. La plage dépend de la CPU.
CD	ZR	BOOL	E, A, M, L, D	Entrée de décrémentation
S	S	BOOL	E, A, M, L, D	Entrée d'initialisation du compteur
PV	ZW	WORD	E, A, M, L, D ou constante	Valeur de comptage entrée sous forme C#<valeur> dans la plage comprise entre 0 et 999
PV	ZW	WORD	E, A, M, L, D	Valeur d'initialisation du compteur
R	R	BOOL	E, A, M, L, D	Entrée de remise à zéro
CV	DUAL	WORD	E, A, M, L, D	Valeur de comptage en cours (format hexadécimal)
CV_BCD	DEZ	WORD	E, A, M, L, D	Valeur de comptage en cours (format DCB)
Q	Q	BOOL	E, A, M, L, D	Etat du compteur

Description de l'opération

Z_RUECK (Paramétrage et compteur de décrémentation)

Un front montant à l'entrée S de cette opération initialise le compteur à la valeur figurant dans l'entrée ZW.

Un 1 à l'entrée R remet le compteur, et donc la valeur de comptage, à zéro.

Le compteur est décrémenté d'une unité si l'état de signal à l'entrée ZR passe de 0 à 1 et que la valeur du compteur 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.

L'état de signal de la sortie Q est à 1 lorsque la valeur de comptage est supérieure à 0 ; il est à 0 lorsque la valeur de comptage est égale à 0.

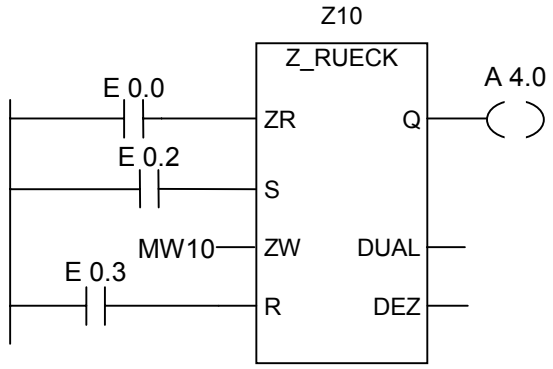
Mot d'état

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

Nota

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

Exemple



Si l'état de signal à l'entrée E 0.2 passe de 0 à 1, le compteur est initialisé à la valeur figurant dans le mot de mémento MW10. Si l'état de signal en 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. L'état de signal de la sortie A 4.0 est 1 si Z10 est différent de zéro.

4.5 ---(SZ) Initialiser compteur

Représentation

Anglaise	Allemande
<C n° >	<Z n° >
---(SC)	**---(SZ)**
<valeur initiale>	<valeur initiale>

Paramètre Anglaise	Paramètre Allemande	Type de données	Zone de mémoire	Description
<C n° >	<Z n° >	COUNTER	Z	Numéro du compteur qui doit être initialisé
<valeur initiale>	<valeur initiale>	WORD	E, A, M, L, D	La valeur d'initialisation (DCB) peut être comprise entre 0 et 999.

Description de l'opération

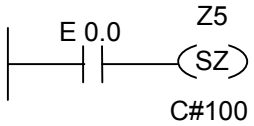
---(SZ) (Initialiser compteur)

Cette opération ne s'exécute que si le RLG présente un front montant. La valeur prédéfinie est alors transférée au compteur indiqué.

Mot d'état

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

Exemple



Le compteur Z5 est initialisé à la valeur 100 si l'état de signal en E 0.0 passe de 0 à 1 (front montant du RLG). En l'absence de front montant, la valeur de Z5 reste inchangée.

4.6 ---(ZV) Incrémenter

Représentation

Anglaise	Allemande
<C n° >	<Z n° >
---(CU)	**---(ZV)**
<valeur initiale>	<valeur initiale>

Paramètre Anglaise	Paramètre Allemande	Type de données	Zone de mémoire	Description
<C n° >	<Z n° >	COUNTER	Z	Numéro du compteur. La plage dépend de la CPU.

Description de l'opération

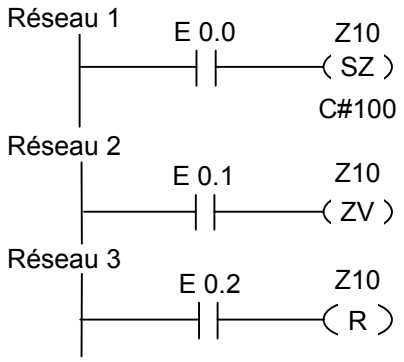
---(ZV) (Incrémenter)

Cette opération incrémente d'un la valeur du compteur précisé si le RLG présente un front montant 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.

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), le compteur Z10 est initialisé avec la valeur 100.

Si l'état de signal en E 0.1 passe de 0 à 1 (front montant du RLG), la valeur de comptage du compteur Z10 est incrémentée d'un, à moins qu'elle ne soit déjà à 999. En l'absence de front montant au RLG, la valeur de Z10 reste inchangée.

Si l'état de signal à l'entrée E 0.2 est égal à 1, le compteur est mis à zéro.

4.7 ---(ZR) Décrémenter

Représentation

Anglaise	**Allemande**
<C n° >	<Z n° >
---(CD)	**---(ZR)**
<valeur initiale>	<valeur initiale>

Paramètre Anglaise	**Paramètre Allemande**	**Type de données**	**Zone de mémoire**	**Description**
<C n° >	<Z n° >	COUNTER	Z	Numéro du compteur. La plage dépend de la CPU.

Description de l'opération

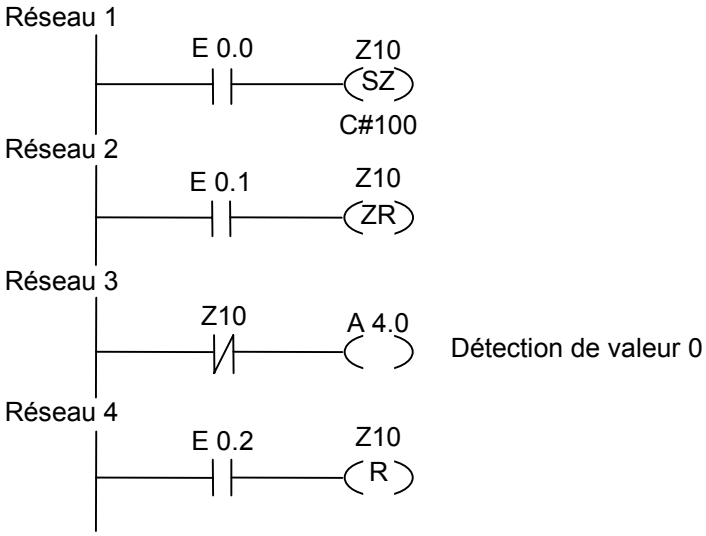
---(ZR) (Décrémenter)

Cette opération décrémente d'un la valeur du compteur précisé si le RLG présente un front montant 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.

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), le compteur Z10 est initialisé avec la valeur 100.

Si l'état de signal en E 0.1 passe de 0 à 1 (front montant du RLG), la valeur de comptage du compteur Z10 est décrémentée d'un, à moins qu'elle ne soit déjà nulle. En l'absence de front montant au RLG, la valeur de Z10 reste inchangée.

Si la valeur de comptage est égale à zéro, la sortie A 4.0 est excitée.

Si l'état de signal à l'entrée E 0.2 est égal à 1, le compteur est mis à zéro.

5 Opérations sur blocs de données

5.1 ---(OPN) Ouvrir bloc de données

Représentation

<DB n°> ou <DI n°>

---(OPN)

Paramètre	Type de données	Zone de mémoire	Description
<DB n°> <DI n°>	BLOCK_DB	DB, DI	Numéro du DB ou du DI. La plage dépend de la CPU.

Description de l'opération

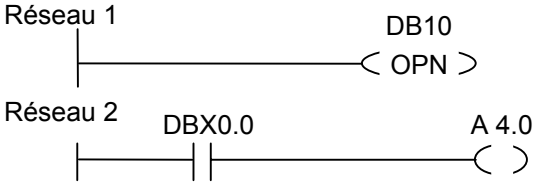
---(OPN) (Ouvrir bloc de données)

Cette opération ouvre un bloc de données (DB) ou un bloc de données d'instance (DI). Il s'agit d'un appel inconditionnel d'un bloc de données. Le numéro du bloc de données est transféré au registre DB ou DI. Les commandes suivantes relatives à des DB et des DI accèdent aux blocs correspondants selon le contenu des registres.

Mot d'état

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

Exemple



Le bloc de données 10 (DB10) est ouvert. L'adresse du contact (DBX0.0) se réfère au bit zéro de l'octet de données zéro de l'enregistrement en cours se trouvant dans le 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
- ---(JMP)--- Saut à l'intérieur d'un bloc si 1 (conditionnel)
- ---(JMPN)--- Saut à l'intérieur d'un bloc si 0 (conditionnel)

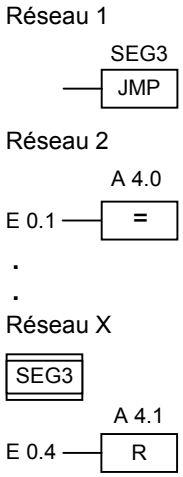
Repère de saut comme opérande

L'opérande d'une opération de saut est un repère de saut qui comporte au maximum 4 caractères. 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 la destination où doit sauter le programme.

Le repère de saut doit être indiqué au-dessus de la bobine de saut.

Repère de saut comme destination

Le repère de destination de saut doit se trouver au début du réseau. Pour l'indiquer, sélectionnez LABEL dans la boîte de sélection CONT. Dans la boîte vide qui apparaît, spécifiez ensuite le nom du repère.



6.2 ---(JMP)--- Saut inconditionnel

Représentation

<repère de saut>

---(JMP)---

Description de l'opération

---(JMP)--- (Saut inconditionnel)

Cette opération fonctionne comme un saut inconditionnel s'il n'y a aucun autre élément CONT entre la barre d'alimentation gauche et l'opération (voir exemple).

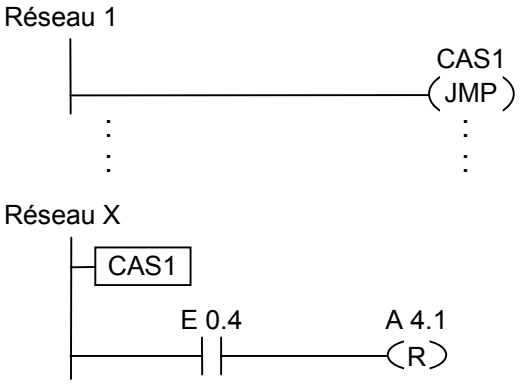
Pour chaque opération ---(JMP)---, il doit exister un repère de saut (LABEL).

Les opérations entre l'opération de saut et le repère de saut ne sont pas exécutées.

Mot d'état

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

Exemple



Le saut est toujours exécuté. Aucune des opérations entre l'opération de saut et le repère de saut n'est traitée.

6.3 ---(JMP)--- Saut à l'intérieur d'un bloc si 1 (conditionnel)

Représentation

<repère de saut>

---(JMP)---

Description de l'opération

---(JMP)--- (Saut à l'intérieur d'un bloc si 1)

Cette opération fonctionne comme un saut conditionnel si le RLG de la combinaison précédente est égal à 1.

Pour chaque opération ---(JMP)---, il doit exister un repère de saut (LABEL).

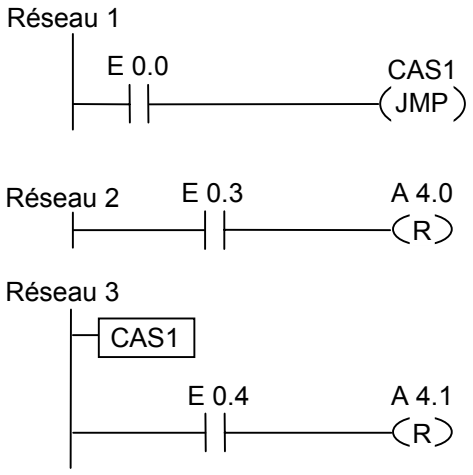
Les opérations entre l'opération de saut et le repère de saut ne sont pas exécutées.

Lorsqu'un saut conditionnel n'est pas exécuté, le RLG passe à 1 après l'opération de saut.

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 à l'intérieur d'un bloc si 0 (conditionnel)

Représentation

<repère de saut>

---(JMPN)

Description de l'opération

---(JMPN) (Saut à l'intérieur d'un bloc si 0)

Cette opération fonctionne comme un saut conditionnel si le RLG de la combinaison précédente est égal à 0.

Pour chaque opération ---(JMPN), il doit exister un repère de saut (LABEL).

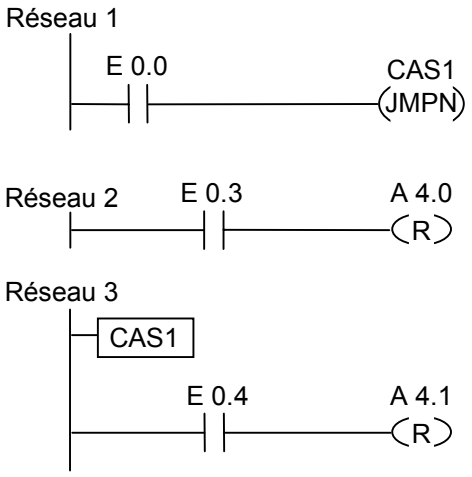
Les opérations entre l'opération de saut et le repère de saut ne sont pas exécutées.

Lorsqu'un saut conditionnel n'est pas exécuté, le RLG passe à 1 après l'opération de saut.

Mot d'état

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

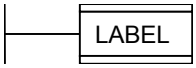
Exemple



Le saut au repère CAS1 est exécuté si l'état de signal est 0 à l'entrée E 0.0. En raison du saut, l'opération de mise à 0 de la sortie A 4.0 n'est pas exécutée même si l'état de signal est 1 à l'entrée E 0.3.

6.5 LABEL Repère de saut

Représentation



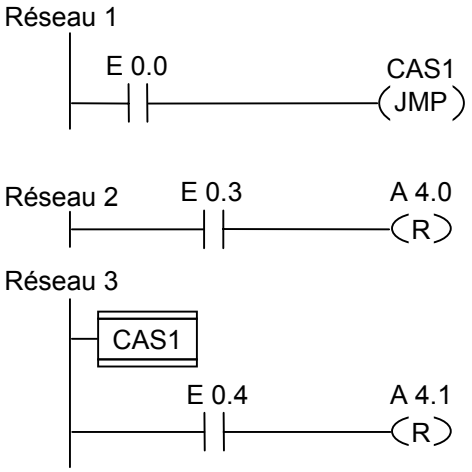
Description

LABEL (repère de saut)

Cette opération 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.

Pour chaque opération ---(JMP) ou ---(JMPN), il doit exister un repère de saut.

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.

7 Fonctions 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 dans les opérations sur nombres entiers

Description

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

- BI1 et BI0,
- DEB,
- 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	*

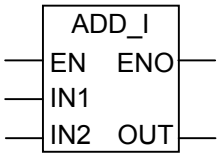
* Le bit DM n'est pas influencé par le résultat de l'opération.

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, L, D	Entrée de validation
ENO	BOOL	E, A, M, L, D	Sortie de validation
IN1	INT	E, A, M, L, D ou constante	Première valeur pour l'addition
IN2	INT	E, A, M, L, D ou constante	Seconde valeur pour l'addition
OUT	INT	E, A, M, L, D	Résultat de l'addition

Description de l'opération

ADD_I (Additionner entiers de 16 bits)

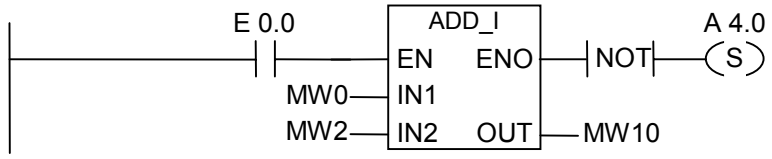
Cette opération additionne les entrées IN1 et IN2 si l'état de signal est 1 à l'entrée de validation EN. Le résultat est rangé dans la sortie OUT. Si ce résultat est hors de la plage autorisée pour un nombre entier de 16 bits, les bits DEB et DM du mot d'état sont mis à 1 et la sortie ENO est à 0. Ainsi, les opérations suivant cette opération arithmétique et qui y sont connectées par ENO (cascade) ne sont pas exécutées.

Voir aussi Evaluation des bits du mot d'état dans 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

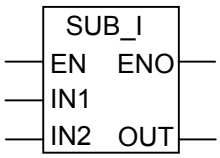
Exemple



L'opération ADD_I est exécutée 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 le résultat est hors de la plage autorisée pour un nombre entier de 16 bits ou si l'état de signal est 0 à l'entrée E 0.0, la sortie A 4.0 est mise à 1.

7.4 SUB_I Soustraire entiers de 16 bits

Représentation



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
IN1	INT	E, A, M, L, D ou constante	Première valeur de la soustraction (de laquelle soustraire)
IN2	INT	E, A, M, L, D ou constante	Valeur à soustraire
OUT	INT	E, A, M, L, D	Résultat de la soustraction

Description de l'opération

SUB_I (Soustraire entiers de 16 bits)

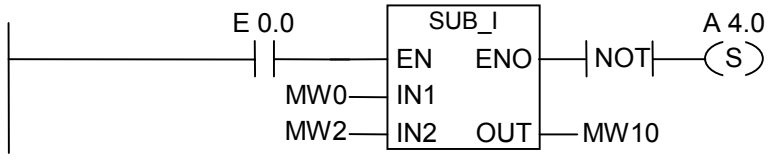
Cette opération soustrait l'entrée IN2 de l'entrée IN1 si l'état de signal est 1 à l'entrée de validation EN. Le résultat est rangé dans la sortie OUT. Si ce résultat est hors de la plage autorisée pour un nombre entier de 16 bits, les bits DEB et DM du mot d'état sont mis à 1 et la sortie ENO est à 0. Ainsi, les opérations suivant cette opération arithmétique et qui y sont connectées par ENO (cascade) ne sont pas exécutées.

Voir aussi Evaluation des bits du mot d'état dans 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

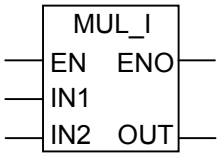
Exemple



L'opération SUB_I est exécutée 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 le résultat est hors de la plage autorisée pour un nombre entier de 16 bits ou si l'état de signal est 0 à l'entrée E 0.0, la sortie A 4.0 est mise à 1.

7.5 MUL_I Multiplier entiers de 16 bits

Représentation



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
IN1	INT	E, A, M, L, D ou constante	Première valeur pour la multiplication
IN2	INT	E, A, M, L, D ou constante	Seconde valeur pour la multiplication
OUT	DINT	E, A, M, L, D	Résultat de la multiplication

Description de l'opération

MUL_I (Multiplier entiers de 16 bits)

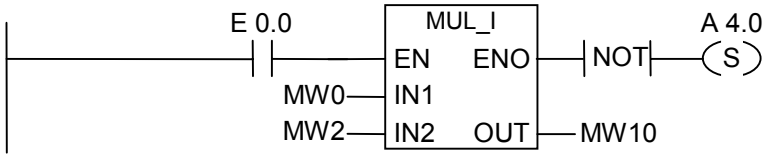
Cette opération multiplie les entrées IN1 et IN2 si l'état de signal est 1 à l'entrée de validation EN. Le résultat est rangé dans la sortie OUT. Si le résultat est hors de la plage autorisée pour un nombre entier de 16 bits, les bits DEB et DM du mot d'état sont mis à 1 et la sortie ENO est à 0. Ainsi, les opérations suivant cette opération arithmétique et qui y sont connectées par ENO (cascade) ne sont pas exécutées.

Voir aussi Evaluation des bits du mot d'état dans 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

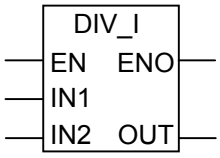
Exemple



L'opération MUL_I est exécutée 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 le résultat est hors de la plage autorisée pour un nombre entier de 16 bits ou si l'état de signal est 0 à l'entrée E 0.0, la sortie A 4.0 est mise à 1.

7.6 DIV_I Diviser entiers de 16 bits

Représentation



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
IN1	INT	E, A, M, L, D ou constante	Dividende
IN2	INT	E, A, M, L, D ou constante	Diviseur
OUT	INT	E, A, M, L, D	Résultat de la division

Description de l'opération

DIV_I (Diviser entiers de 16 bits)

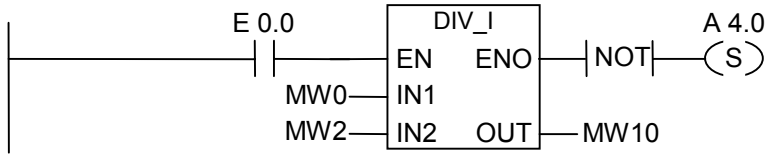
Cette opération divise l'entrée IN1 par l'entrée IN2 si l'état de signal est 1 à l'entrée de validation EN. Le résultat est rangé dans la sortie OUT. Si le résultat est hors de la plage autorisée pour un nombre entier de 16 bits, les bits DEB et DM du mot d'état sont mis à 1 et la sortie ENO est à 0. Ainsi, les opérations suivant cette opération arithmétique et qui y sont connectées par ENO (cascade) ne sont pas exécutées.

Voir aussi Evaluation des bits du mot d'état dans 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

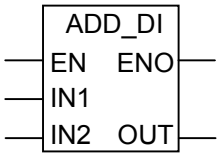
Exemple



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

7.7 ADD_DI Additionner entiers de 32 bits

Représentation



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
IN1	DINT	E, A, M, L, D ou constante	Première valeur pour l'addition
IN2	DINT	E, A, M, L, D ou constante	Seconde valeur pour l'addition
OUT	DINT	E, A, M, L, D	Résultat de l'addition

Description de l'opération

ADD_DI (Additionner entiers de 32 bits)

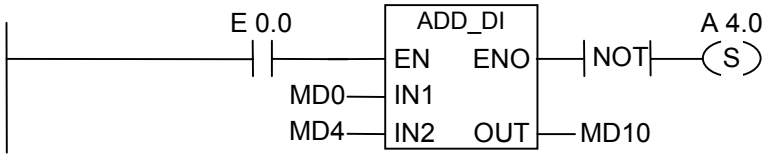
Cette opération additionne IN1 et IN2 si l'état de signal est 1 à l'entrée de validation EN. Le résultat est rangé dans la sortie OUT. Si ce résultat est hors de la plage autorisée pour un nombre entier de 32 bits, les bits DEB et DM du mot d'état sont mis à 1 et la sortie ENO est à 0. Ainsi, les opérations suivant cette opération arithmétique et qui y sont connectées par ENO (cascade) ne sont pas exécutées.

Voir aussi Evaluation des bits du mot d'état dans 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

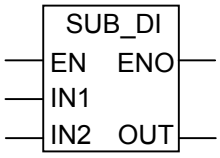
Exemple



L'opération ADD_DI est exécutée 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 le résultat est hors de la plage autorisée pour un nombre entier de 32 bits ou si l'état de signal est 0 à l'entrée E 0.0, la sortie A 4.0 est mise à 1.

7.8 SUB_DI Soustraire entiers de 32 bits

Représentation



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
IN1	DINT	E, A, M, L, D ou constante	Première valeur de la soustraction
IN2	DINT	E, A, M, L, D ou constante	Valeur à soustraire
OUT	DINT	E, A, M, L, D	Résultat de la soustraction

Description de l'opération

SUB_DI (Soustraire entiers de 32 bits)

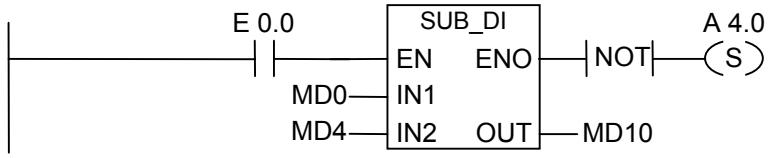
Cette opération soustrait l'entrée IN2 de l'entrée IN1 si l'état de signal est 1 à l'entrée de validation EN. Le résultat est rangé dans la sortie OUT. Si ce résultat est hors de la plage autorisée pour un nombre entier de 32 bits, les bits DEB et DM du mot d'état sont mis à 1 et la sortie ENO est à 0. Ainsi, les opérations suivant cette opération arithmétique et qui y sont connectées par ENO (cascade) ne sont pas exécutées.

Voir aussi Evaluation des bits du mot d'état dans 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

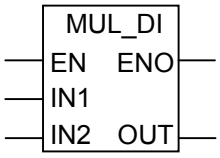
Exemple



L'opération SUB_DI est exécutée 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 le résultat est hors de la plage autorisée pour un nombre entier de 32 bits ou si l'état de signal est 0 à l'entrée E 0.0, la sortie A 4.0 est mise à 1.

7.9 MUL_DI Multiplier entiers de 32 bits

Représentation



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
IN1	DINT	E, A, M, L, D ou constante	Première valeur pour la multiplication
IN2	DINT	E, A, M, L, D ou constante	Seconde valeur pour la multiplication
OUT	DINT	E, A, M, L, D	Résultat de la multiplication

Description de l'opération

MUL_DI (Multiplier entiers de 32 bits)

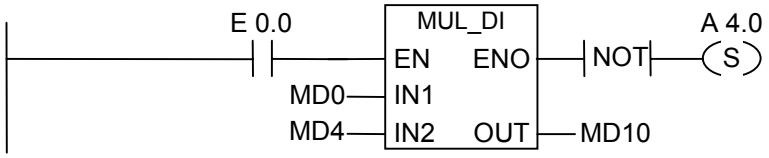
Cette opération multiplie les entrées IN1 et IN2 si l'état de signal est 1 à l'entrée de validation EN. Le résultat est rangé dans la sortie OUT. Si le résultat est hors de la plage autorisée pour un nombre entier de 32 bits, les bits DEB et DM du mot d'état sont mis à 1 et la sortie ENO est à 0. Ainsi, les opérations suivant cette opération arithmétique et qui y sont connectées par ENO (cascade) ne sont pas exécutées.

Voir aussi Evaluation des bits du mot d'état dans 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

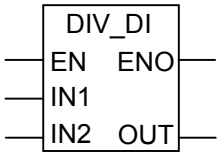
Exemple



L'opération MUL_DI est exécutée 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 le résultat est hors de la plage autorisée pour un nombre entier de 32 bits ou si l'état de signal est 0 à l'entrée E 0.0, la sortie A 4.0 est mise à 1.

7.10 DIV_DI Diviser entiers de 32 bits

Représentation



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
IN1	DINT	E, A, M, L, D ou constante	Dividende
IN2	DINT	E, A, M, L, D ou constante	Diviseur
OUT	DINT	E, A, M, L, D	Résultat entier de la division

Description de l'opération

DIV_DI (Diviser entiers de 32 bits)

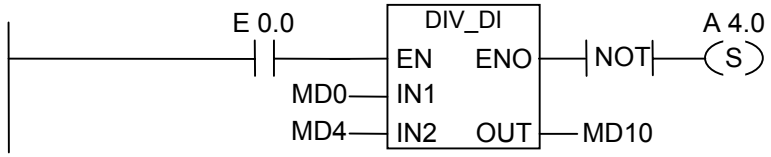
Cette opération divise l'entrée IN1 par l'entrée IN2 si l'état de signal est 1 à l'entrée de validation EN. Le résultat (partie entière) est rangé dans la sortie OUT. Cette opération ne fournit pas de reste. Si le quotient est hors de la plage autorisée pour un nombre entier de 32 bits, les bits DEB et DM du mot d'état sont mis à 1 et la sortie ENO est à 0. Ainsi, les opérations suivant cette opération arithmétique et qui y sont connectées par ENO (cascade) ne sont pas exécutées.

Voir aussi Evaluation des bits du mot d'état dans 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

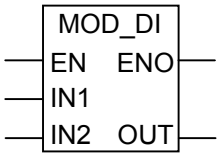
Exemple



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

7.11 MOD_DI Reste de division (32 bits)

Représentation



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
IN1	DINT	E, A, M, L, D ou constante	Dividende
IN2	DINT	E, A, M, L, D ou constante	Diviseur
OUT	DINT	E, A, M, L, D	Reste de la division

Description de l'opération

MOD_DI (Reste de division)

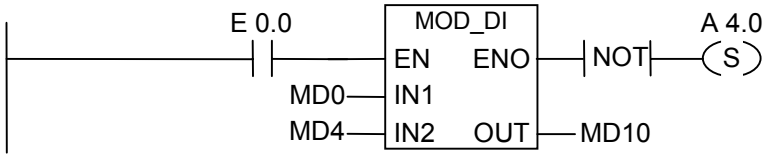
Cette opération divise l'entrée IN1 par l'entrée IN2 si l'état de signal est 1 à l'entrée de validation EN. Le reste de la division est rangé dans la sortie OUT. Si ce reste est hors de la plage autorisée pour un nombre entier de 32 bits, les bits DEB et DM du mot d'état sont mis à 1 et la sortie ENO est à 0. Ainsi, les opérations suivant cette opération arithmétique et qui y sont connectées par ENO (cascade) ne sont pas exécutées.

Voir aussi Evaluation des bits du mot d'état dans 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



L'opération MOD_DI est exécutée 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 est 0 à l'entrée E 0.0, la sortie A 4.0 est mise à 1.

8 Fonctions sur nombres à virgule flottante

8.1 Vue d'ensemble des opérations arithmétiques sur nombres à virgule flottante

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 Addition
- SUB_R Soustraction
- MUL_R Multiplication
- DIV_R Division

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) ou Racine carrée (SQRT) d'un nombre réel
- Logarithme naturel (LN) d'un nombre réel
- Valeur exponentielle (EXP) sur la base e (= 2,71828) d'un nombre réel
- Fonctions trigonométriques d'angles représentés sous forme de nombres réels IEEE de 32 bits :
  - Sinus (SIN) et Arc sinus (ASIN)
  - Cosinus (COS) et Arc cosinus (ACOS)
  - Tangente (TAN) et Arc tangente (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 dans les opérations sur nombres à virgule flottante

Description

Les opérations arithmétiques sur nombres à virgule flottante affectent les bits suivants du mot d'état :

- BI1 et BI0,
- DEB,
- 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	*

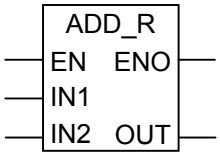
* Le bit DM n'est pas influencé par le résultat de l'opération.

Plage incorrecte	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 réels

Représentation



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
IN1	REAL	E, A, M, L, D ou constante	Première valeur pour l'addition
IN2	REAL	E, A, M, L, D ou constante	Seconde valeur pour l'addition
OUT	REAL	E, A, M, L, D	Résultat de l'addition

Description de l'opération

ADD_R (Additionner réels)

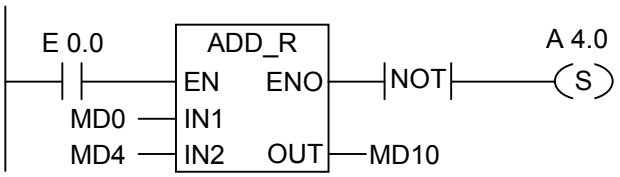
Si l'état de signal est 1 à l'entrée de validation (EN), cette opération additionne les entrées IN1 et IN2 et range le résultat dans la sortie OUT. Si ce résultat est hors de la plage autorisée pour un nombre réel (débordement ou dépassement bas), les bits DEB et DM du mot d'état sont mis à 1 et ENO est mis à 0. Ainsi, les opérations suivant cette opération arithmétique et qui y sont connectées par ENO (cascade) ne sont pas exécutées.

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

Mot d'état

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

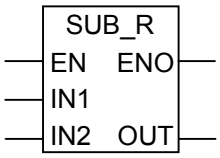
Exemple



L'opération ADD_R est exécutée 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 ce résultat est hors de la plage autorisée pour un nombre réel ou si cette addition n'est pas traitée, la sortie A 4.0 est mise à 1.

8.3.2 SUB_R Soustraire réels

Représentation



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
IN1	REAL	E, A, M, L, D ou constante	Première valeur de soustraction (de laquelle soustraire)
IN2	REAL	E, A, M, L, D ou constante	Valeur à soustraire de la première valeur
OUT	REAL	E, A, M, L, D	Résultat de la soustraction

Description de l'opération

SUB_R (Soustraire réels)

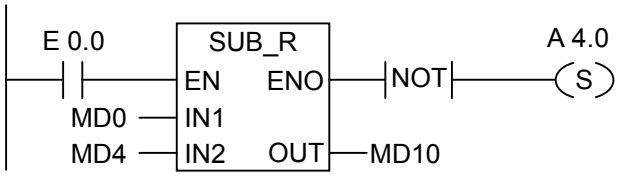
Si l'état de signal est 1 à l'entrée de validation (EN), cette opération soustrait l'entrée IN2 de l'entrée IN1 et range le résultat dans la sortie OUT. Si ce résultat est hors de la plage autorisée pour un nombre réel (débordement ou dépassement bas), les bits DEB et DM du mot d'état sont mis à 1 et ENO est mis à 0. Ainsi, les opérations suivant cette opération arithmétique et qui y sont connectées par ENO (cascade) ne sont pas exécutées.

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

Mot d'état

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

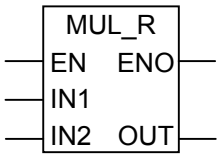
Exemple



L'opération SUB_R est exécutée si l'état de signal est à 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 réel ou si cette soustraction n'est pas traitée, la sortie A 4.0 est mise à 1.

8.3.3 MUL_R Multiplier réels

Représentation



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
IN1	REAL	E, A, M, L, D ou constante	Première valeur pour la multiplication
IN2	REAL	E, A, M, L, D ou constante	Seconde valeur pour la multiplication
OUT	REAL	E, A, M, L, D	Résultat de la multiplication

Description de l'opération

MUL_R (Multiplier réels)

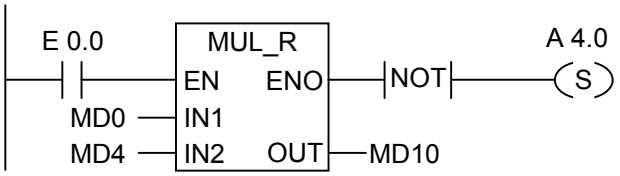
Si l'état de signal est 1 à l'entrée de validation (EN), cette opération multiplie les entrées IN1 et IN2 et range le résultat dans la sortie OUT. Si ce résultat est hors de la plage autorisée pour un nombre réel (débordement ou dépassement bas), les bits DEB et DM du mot d'état sont mis à 1 et ENO est mis à 0. Ainsi, les opérations suivant cette opération arithmétique et qui y sont connectées par ENO (cascade) ne sont pas exécutées.

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

Mot d'état

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

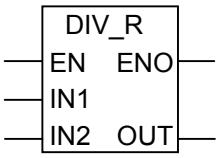
Exemple



L'opération MUL_R est exécutée 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 le résultat est hors de la plage autorisée pour un nombre réel ou si cette multiplication n'est pas traitée, la sortie A 4.0 est mise à 1.

8.3.4 DIV_R Diviser réels

Représentation



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
IN1	REAL	E, A, M, L, D ou constante	Dividende
IN2	REAL	E, A, M, L, D ou constante	Diviseur
OUT	REAL	E, A, M, L, D	Résultat de la division

Description de l'opération

DIV_R (Diviser réels)

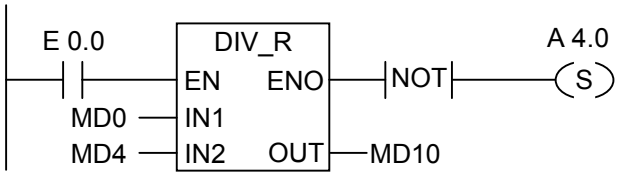
Si l'état de signal est 1 à l'entrée de validation (EN), cette opération divise l'entrée IN1 par l'entrée IN2 et range le résultat dans la sortie OUT. Si ce résultat est hors de la plage autorisée pour un nombre réel (débordement ou dépassement bas), les bits DEB et DM du mot d'état sont mis à 1 et ENO est mis à 0. Ainsi, les opérations suivant cette opération arithmétique et qui y sont connectées par ENO (cascade) ne sont pas exécutées.

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

Mot d'état

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

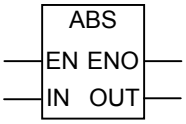
Exemple



L'opération DIV_R est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le résultat de la division MD0 par MD4 est rangé dans le double mot de mémento MD10. Si ce résultat est hors de la plage autorisée pour un nombre réel ou si cette division n'est pas traitée, la sortie A 4.0 est mise à 1.

8.3.5 ABS Valeur absolue d'un nombre réel

Représentation



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
IN	REAL	E, A, M, L, D ou constante	Valeur d'entrée : nombre réel
OUT	REAL	E, A, M, L, D	Valeur de sortie : valeur absolue du nombre réel

Description de l'opération

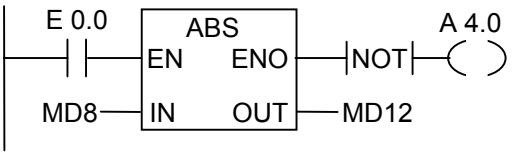
ABS (Valeur absolue d'un nombre réel)

Cette opération forme la valeur absolue d'un nombre réel.

Mot d'état

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

Exemple



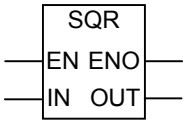
Si l'état de signal est 1 à l'entrée E 0.0, MD8 transmet la valeur absolue à MD12.

De MD8 = -6,234 résulte MD12 = +6,234. Si la conversion n'est pas exécutée, l'état de signal est 1 à la sortie A 4.0 (ENO = EN = 0).

8.4 Opérations étendues

8.4.1 SQR Carré

Représentation



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
IN	REAL	E, A, M, L, D ou constante	Valeur d'entrée : nombre réel
OUT	REAL	E, A, M, L, D	Valeur de sortie : carré du nombre réel

Description de l'opération

L'opération **SQR** (Carré d'un nombre réel) calcule le carré d'un nombre réel.

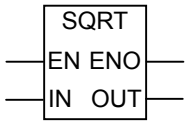
Voir aussi Evaluation des bits du mot d'état dans les opérations sur nombres à virgule flottante.

Mot d'état

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

8.4.2 SQRT Racine carrée

Représentation



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
IN	REAL	E, A, M, L, D ou constante	Valeur d'entrée : nombre réel
OUT	REAL	E, A, M, L, D	Valeur de sortie : racine carrée du nombre réel

Description de l'opération

L'opération **SQRT** (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 0. Unique exception : la racine carrée de -0 est -0.

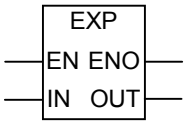
Voir aussi Evaluation des bits du mot d'état dans les opérations sur nombres à virgule flottante.

Mot d'état

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

8.4.3 EXP Valeur exponentielle

Représentation



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
IN	REAL	E, A, M, L, D ou constante	Valeur d'entrée : nombre réel
OUT	REAL	E, A, M, L, D	Valeur de sortie : valeur exponentielle du nombre réel

Description de l'opération

L'opération **EXP** (Valeur exponentielle d'un nombre réel) calcule la valeur exponentielle de base e (= 2,71828...) d'un nombre réel.

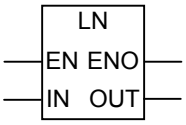
Voir aussi Evaluation des bits du mot d'état dans les opérations sur nombres à virgule flottante.

Mot d'état

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

8.4.4 LN Logarithme naturel

Représentation



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
IN	REAL	E, A, M, L, D ou constante	Valeur d'entrée : nombre réel
OUT	REAL	E, A, M, L, D	Valeur de sortie : logarithme naturel du nombre réel

Description de l'opération

L'opération **LN** (Logarithme naturel d'un nombre réel) calcule le logarithme naturel d'un nombre réel.

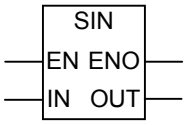
Voir aussi Evaluation des bits du mot d'état dans les opérations sur nombres à virgule flottante.

Mot d'état

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

8.4.5 SIN Sinus

Représentation



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
IN	REAL	E, A, M, L, D ou constante	Valeur d'entrée : nombre réel
OUT	REAL	E, A, M, L, D	Valeur de sortie : sinus du nombre réel

Description de l'opération

L'opération **SIN** (Sinus d'un nombre réel) calcule le sinus d'un nombre réel qui représente un angle en radians.

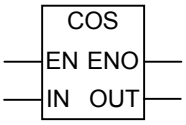
Voir aussi Evaluation des bits du mot d'état dans les opérations sur nombres à virgule flottante.

Mot d'état

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

8.4.6 COS Cosinus

Représentation



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
IN	REAL	E, A, M, L, D ou constante	Valeur d'entrée : nombre réel
OUT	REAL	E, A, M, L, D	Valeur de sortie : cosinus du nombre réel

Description de l'opération

L'opération **COS** (Cosinus d'un nombre réel) calcule le cosinus d'un nombre réel qui représente un angle en radians.

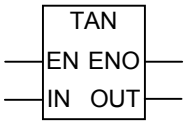
Voir aussi Evaluation des bits du mot d'état dans les opérations sur nombres à virgule flottante.

Mot d'état

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

8.4.7 TAN Tangente

Représentation



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
IN	REAL	E, A, M, L, D ou constante	Valeur d'entrée : nombre réel
OUT	REAL	E, A, M, L, D	Valeur de sortie : tangente du nombre réel

Description de l'opération

L'opération **TAN** (Tangente d'un nombre réel) calcule la tangente d'un nombre réel qui représente un angle en radians.

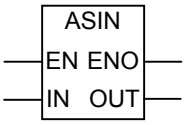
Voir aussi Evaluation des bits du mot d'état dans les opérations sur nombres à virgule flottante.

Mot d'état

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

8.4.8 ASIN Arc sinus

Représentation



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
IN	REAL	E, A, M, L, D ou constante	Valeur d'entrée : nombre réel
OUT	REAL	E, A, M, L, D	Valeur de sortie : arc sinus d'un nombre réel

Description de l'opération

L'opération **ASIN** (Arc sinus d'un nombre réel) calcule l'arc sinus d'un nombre réel dont la valeur d'entrée doit être comprise entre :

-1 <= valeur d'entrée <= +1

Le résultat est un angle indiqué en radians. Sa valeur est comprise dans la plage suivante :

$-\pi/2 \leq$ valeur de sortie $\leq +\pi/2$

avec π = 3,1415....

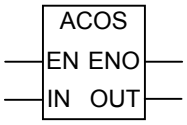
Voir aussi Evaluation des bits du mot d'état dans les opérations sur nombres à virgule flottante.

Mot d'état

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

8.4.9 ACOS Arc cosinus

Représentation



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
IN	REAL	E, A, M, L, D ou constante	Valeur d'entrée : nombre réel
OUT	REAL	E, A, M, L, D	Valeur de sortie : arc cosinus du nombre réel

Description de l'opération

L'opération **ACOS** (Arc cosinus d'un nombre réel) calcule l'arc cosinus d'un nombre réel dont la valeur d'entrée doit être comprise entre :

-1 <= valeur d'entrée <= +1

Le résultat est un angle indiqué en radians. Sa valeur est comprise dans la plage suivante :

$0 \leq$ valeur de sortie $\leq +\pi$

avec π = 3,1415....

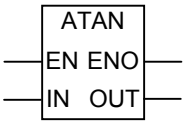
Voir aussi Evaluation des bits du mot d'état dans les opérations sur nombres à virgule flottante.

Mot d'état

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

8.4.10 ATAN Arc tangente

Représentation



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
IN	REAL	E, A, M, L, D ou constante	Valeur d'entrée : nombre réel
OUT	REAL	E, A, M, L, D	Valeur de sortie : arc tangente du nombre réel

Description de l'opération

L'opération **ATAN** (Arc tangente d'un nombre réel) calcule l'arc tangente d'un nombre réel. Le résultat est un angle en radians dont la valeur est comprise dans la plage suivante :

$$-\pi/2 \leq \text{valeur de sortie} \leq +\pi/2$$

avec π = 3,1415....

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

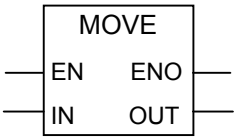
Mot d'état

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

9 Opérations de transfert

9.1 MOVE Affecter valeur

Représentation



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
IN	Tous les types de données simples d'une longueur de 8, de 16 ou de 32 bits	E, A, M, L, D ou constante	Valeur source
OUT	Tous les types de données élémentaires d'une longueur de 8, de 16 ou de 32 bits	E, A, M, L, D	Adresse de destination

Description de l'opération

MOVE (Affecter valeur)

Cette opération est activée par l'entrée de validation EN. La valeur indiquée dans l'entrée IN est copiée à l'adresse précisée dans la sortie OUT. L'état de signal de ENO est identique à celui de EN. L'opération **MOVE** ne permet de copier que des octets, des mots ou des doubles mots. Pour copier des types de données utilisateur tels que des tableaux ou des structures, vous devez faire appel à la fonction système "BLKMOV" (SFC 20).

Mot d'état

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

Dépendance par rapport au relais de masquage (Master Control Relay, MCR)

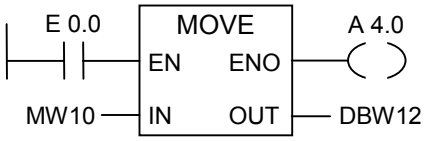
La dépendance par rapport au relais MCR est uniquement activée lorsque l'opération MOVE est à l'intérieur d'une zone MCR active. Au sein d'une telle zone active, les données adressées sont copiées comme décrit ci-dessus si le relais MCR est en fonction et que le flux d'énergie est présent à l'entrée de validation. Si le relais MCR est hors fonction et si une opération MOVE est exécutée, la valeur 0 est écrite à l'adresse précisée dans la sortie OUT, quel que soit l'état de signal en vigueur à l'entrée IN.

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. Exemples :

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
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 E 0.0 est à 1. Le contenu de MW10 est alors copié dans le mot de données 12 du bloc de données en cours.

La sortie A 4.0 est mise à 1 si l'opération est exécutée.

Si le trajet de courant de l'exemple est à l'intérieur d'une zone MCR active :

Si le relais MCR est en fonction, les données sont copiées comme décrit ci-dessus de MW10 dans DBW12.

Si le relais MCR est hors fonction, la valeur 0 est écrite dans le DBW12.

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 (boîte)
- CALL_FC Appeler FC (boîte)
- CALL_SFB Appeler SFB (boîte)
- CALL_SFC Appeler SFC (boîte)
- Appeler multi-instance
- Appeler un bloc dans une bibliothèque

- ---(MCR<) Relais de masquage en fonction
- ---(MCR>) Relais de masquage hors fonction
- ---(MCRA) Activer relais de masquage
- ---(MCRD) Désactiver relais de masquage

- RET Retour

Voir aussi Remarques importantes sur l'utilisation de la fonctionnalité MCR

10.2 ---(Call) Appeler FC/SFC sans paramètre

Représentation

<FC/SFC n°>

---(CALL)

Paramètre	Type de données	Zone de mémoire	Description
<FC/SFC n°>	BLOCK_FC	-	Numéro de la fonction ou de la fonction système. La plage dépend de la CPU.

Description de l'opération

---(Call) (Appeler FC/SFC sans paramètre)

Cette opération permet d'appeler une fonction (FC) ou une fonction système (SFC) qui n'a pas de paramètre. L'appel est uniquement exécuté lorsque le RLG est 1 à la bobine CALL. Si l'opération ---(CALL) a lieu, elle fonctionne comme suit :

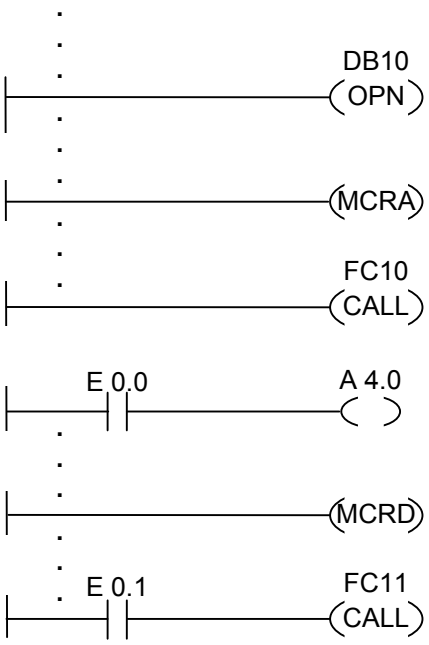
- Elle sauvegarde l'adresse de retour au bloc appelant.
- Elle change la zone de données locales en cours en zone de données locales précédente.
- Elle empile le bit MA (bit MCR actif) dans la pile des blocs.
- Elle crée la nouvelle zone de données locales pour la fonction appelée.

Ensuite, le traitement du programme se poursuit dans la fonction ou dans la fonction système appelée.

Mot d'état

		RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Appel conditionnel :	Ecriture	-	-	-	-	0	0	1	1	0
Appel inconditionnel :	Ecriture	-	-	-	-	0	0	1	-	0

Exemple



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 de la FC10 est exécuté, voici ce qui se passe :

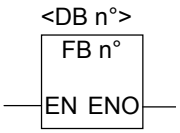
L'adresse de retour au bloc fonctionnel appelant et les sélecteurs pour le DB10 et pour le bloc de données d'instance du FB appelant sont sauvegardés. Le bit MA qui a été mis à 1 par l'opération MCRA est empilé dans la pile des blocs, puis mis à 0 pour la fonction FC10 appelée. Le traitement du programme 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 (DI) du FB utilisateur redeviennent les DB en cours. Le programme se poursuit avec l'opération suivante : l'affectation de l'état de signal en E 0.0 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, la gestion du programme est transmise à la FC11 comme décrit pour la FC10 et revient de la FC11 après traitement.

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.

10.3 CALL_FB Appeler FB (boîte)

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 n° DB n°	BLOCK_FB BLOCK_DB	- -	Numéros du FB et du DB. La plage dépend de la CPU.

Description de l'opération

CALL_FB (Appeler FB)

Cette opération est exécutée si EN est à 1. Si l'opération **CALL_FB** a lieu, elle fonctionne comme suit :

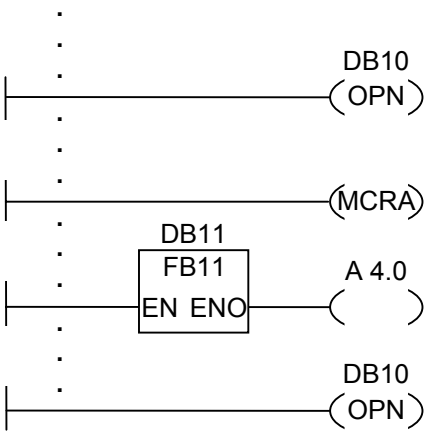
- Elle sauvegarde l'adresse de retour au bloc appelant.
- Elle sauvegarde les sélecteurs pour les deux blocs de données en cours (DB et DI).
- Elle change la zone de données locales en cours en zone de données locales précédente.
- Elle empile le bit MA (bit MCR actif) dans la pile des blocs.
- Elle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

Ensuite, le traitement du programme se poursuit dans le bloc fonctionnel 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).

Mot d'état

		RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Appel conditionnel :	Ecriture	x	-	-	-	0	0	x	x	x
Appel inconditionnel :	Ecriture	-	-	-	-	0	0	x	x	x

Exemple



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 FB11 est exécuté, voici ce qui se passe :

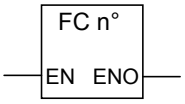
L'adresse de retour au bloc fonctionnel appelant et les sélecteurs pour le DB10 et pour le bloc de données d'instance de ce bloc fonctionnel sont sauvegardés. Le bit MA qui a été mis à 1 par l'opération MCRA est empilé dans la pile des blocs, puis mis à 0 pour le bloc FB11 appelé. Ensuite, le traitement 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 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. Le DB requis doit être de nouveau ouvert.

10.4 CALL_FC Appeler FC (boîte)

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 n°	BLOCK_FC	-	Numéro de la FC. La plage dépend de la CPU.

Description de l'opération

CALL_FC (Appeler FC)

Cette opération permet d'appeler une fonction (FC). L'appel est activé par l'état de signal 1 à l'entrée de validation EN. 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 la fonction appelée.

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).

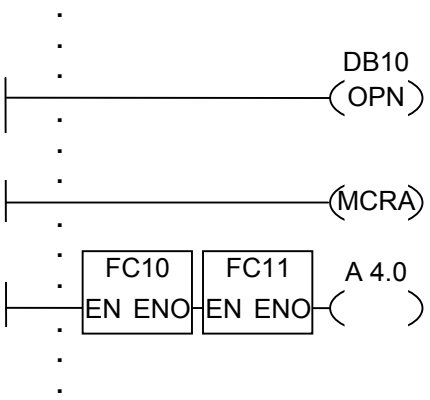
Lorsque vous appelez une fonction (FC) et que la table de déclaration des variables du bloc appelé comporte des déclarations du type IN, OUT et IN_OUT, ces variables s'affichent sous forme de liste de paramètres formels dans le programme du bloc appelant.

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
Appel conditionnel :	Ecriture	x	-	-	-	0	0	x	x	x
Appel inconditionnel :	Ecriture	-	-	-	-	0	0	x	x	x

Exemple



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 de la FC10 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 FC10. Ensuite, le traitement du programme se poursuit dans la fonction 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 via 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 avec l'opération suivante dont l'exécution est fonction de 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.

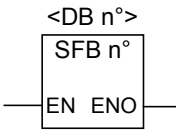
Si la FC11 est également exécutée sans erreur, ENO et donc A 4.0 sont à 1.

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 (boîte)

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 n° DB n°	BLOCK_SFB BLOCK_DB	- -	Numéro du SFB et du DB. La plage dépend de la CPU.

Description de l'opération

CALL_SFB (Appeler SFB)

Cette opération est exécutée si EN est à 1. Si l'opération a lieu, elle fonctionne comme suit :

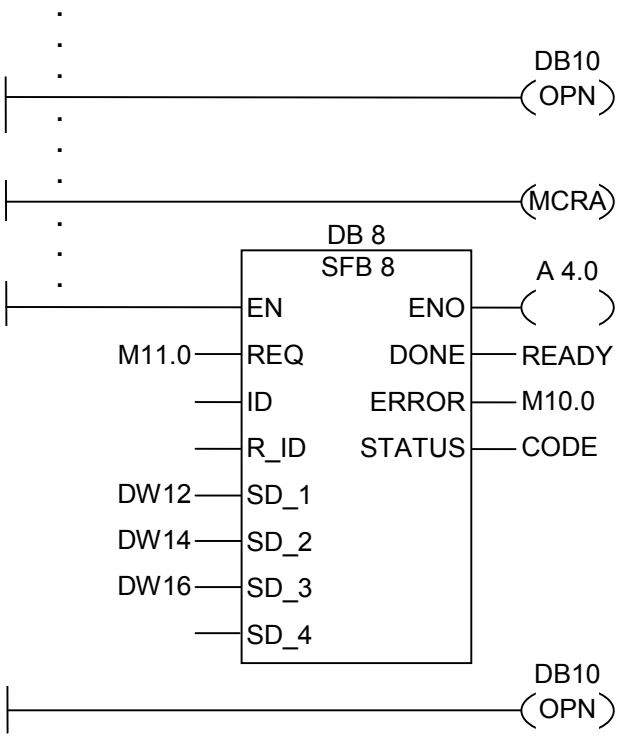
- Elle sauvegarde l'adresse de retour au bloc appelant.
- Elle sauvegarde les sélecteurs pour les deux blocs de données en cours (DB et DI).
- Elle change la zone de données locales en cours en zone de données locales précédente.
- Elle empile le bit MA (bit MCR actif) dans la pile des blocs.
- Elle 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
Appel conditionnel :	Ecriture	x	-	-	-	0	0	x	x	x
Appel inconditionnel :	Ecriture	-	-	-	-	0	0	x	x	x

Exemple



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 (boîte)

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 n°	BLOCK_SFC	-	Numéro de la SFC. La plage dépend de la CPU.

Description de l'opération

CALL_SFC (Appeler SFC)

Cette opération permet d'appeler une fonction système. L'appel est activé par l'état de signal 1 à l'entrée de validation EN. Si l'opération a lieu, elle fonctionne comme suit :

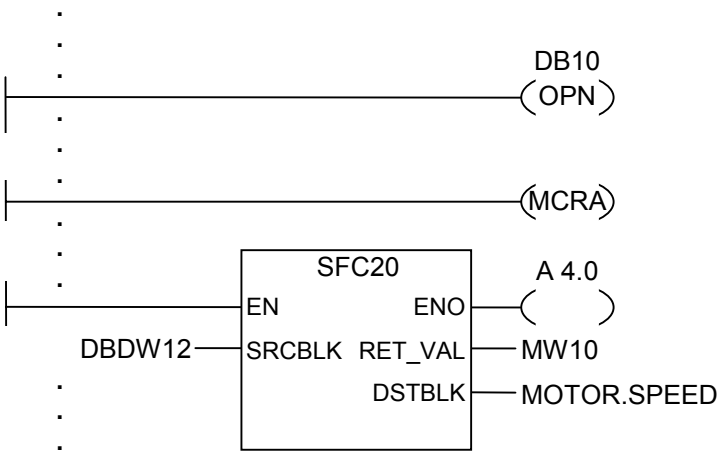
- Elle sauvegarde l'adresse de retour au bloc appelant.
- Elle change la zone de données locales en cours en zone de données locales précédente.
- Elle empile le bit MA (bit MCR actif) dans la pile des blocs.
- Elle crée la nouvelle zone de données locales pour la fonction appelée.

Ensuite, le traitement du programme se poursuit dans la fonction système appelée. 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
Appel conditionnel :	Ecriture	x	-	-	-	0	0	x	x	x
Appel inconditionnel :	Ecriture	-	-	-	-	0	0	x	x	x

Exemple



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 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

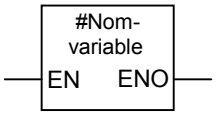
ENO = 0 A 4.0 = 0

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-instance

Représentation



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
#Nom-variable	FB, SFB	-	Nom de la multi-instance

Description de l'opération

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 varie selon les paramètres existants et leur nombre. EN, ENO et le nom de 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 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 ---(MCR<) et ---(MCR>) ! Ceci concerne alors évidemment aussi **toutes** les boîtes contenant une affectation, y compris la transmission de paramètres à des blocs !
- Le MCR se trouve précisément hors fonction lorsque le RLG était égal à 0 avant une instruction **MCR<**.

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.

10.10 ---(MCR<) Relais de masquage en fonction

Remarques importantes sur l'utilisation de la fonctionnalité MCR

Représentation

---(MCR<)

Description de l'opération

---(MCR<) (Relais de masquage en fonction)

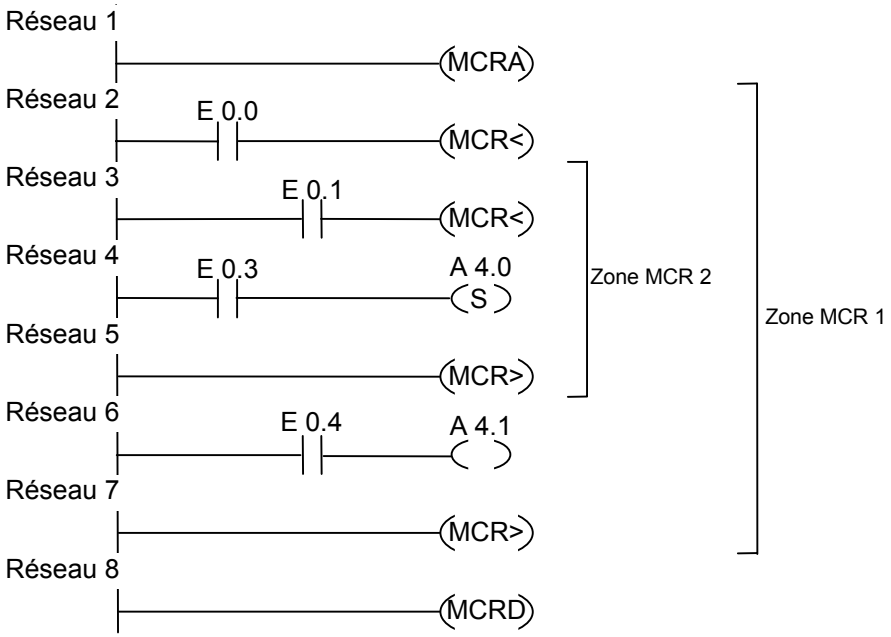
Cette opération empile le résultat logique RLG dans la pile MCR et ouvre une zone MCR. La pile MCR fonctionne selon le principe "dernier entré, premier sorti" (LIFO) et peut contenir jusqu'à huit entrées (8 niveaux). Si elle est pleine, l'opération ---(MCR<) provoque une erreur de pile MCR (MCRF). Les éléments suivants dépendent du relais MCR et sont influencés par l'état de signal du RLG qui est empilé dans la pile MCR tant qu'une zone MCR est ouverte :

- --(#) Connecteur
- --( ) Bobine de sortie
- --(S) Mettre à 1
- --(R) Mettre à 0
- RS Bascule mise à 0, mise à 1
- SR Bascule mise à 1, mise à 0
- MOVE Affecter valeur

Mot d'état

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

Exemple



L'opération MCRA active la fonction MCR. Ensuite, vous pouvez créer jusqu'à huit zones MCR imbriquées. Dans notre exemple, il y en a deux. Les opérations sont exécutées comme suit :

Si l'entrée E 0.0 est à 1 (le relais MCR est **en fonction** dans la zone 1), l'état de signal à l'entrée E 0.4 est affecté à la sortie A 4.1.

Si l'entrée E 0.0 est à 0 (le MCR est **hors fonction** dans la zone 1), l'état de signal de la sortie A 4.1 est 0, quel que soit l'état de signal à l'entrée E 0.4.

Si les entrées E 0.0 et E 0.1 sont à 1 (le relais MCR est **en fonction** dans la zone 2), la sortie A 4.0 est mise à 1 si l'état de signal à l'entrée E 0.3 est à 1.

Si les entrées E 0.0 ET E 0.1 sont à 0 (le relais MCR est **hors fonction** dans la zone 2), la sortie A 4.0 reste inchangée, quel que soit l'état de signal à l'entrée E 0.3.

10.11 ---(MCR>) Relais de masquage hors fonction

Remarques importantes sur l'utilisation de la fonctionnalité MCR

Représentation

---(MCR>)

Description de l'opération

---(MCR>) (Relais de masquage hors fonction)

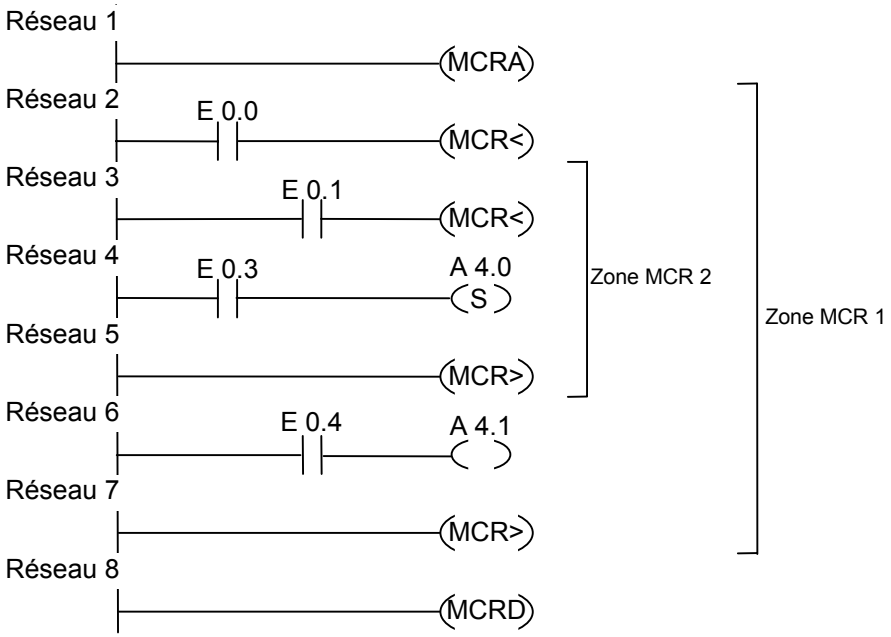
Cette opération retire de la pile MCR une entrée RLG qui y avait été empilée. La pile MCR fonctionne selon le principe "dernier entré, premier sorti" (LIFO) et peut enregistrer jusqu'à huit entrées (8 niveaux). Si la pile est vide, l'opération ---(MCR>) provoque une erreur de pile MCR (MCRF). Les éléments suivants dépendent du relais MCR et sont influencés par l'état de signal du RLG empilé dans la pile MCR tant qu'une zone MCR est ouverte :

- --(#) Connecteur
- --( ) Bobine de sortie
- --(S) Mettre à 1
- --(R) Mettre à 0
- RS Bascule mise à 0, mise à 1
- SR Bascule mise à 1, mise à 0
- MOVE Affecter valeur

Mot d'état

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

Exemple



L'opération ---(MCRA) active la fonction MCR. Ensuite, vous pouvez créer jusqu'à huit zones MCR. Dans notre exemple, il y en a deux. La première opération ---(MCR>) (relais MCR hors fonction) va de pair avec la seconde opération ---(MCR<) (relais MCR en fonction). Toutes les fonctions entre ces deux opérations appartiennent à la seconde zone MCR. Les fonctions sont exécutées de la manière suivante :

Si l'entrée E 0.0 est à 1, l'état de signal à l'entrée E 0.4 est affecté à la sortie A 4.1.

Si l'entrée E 0.0 est à 0, la sortie A 4.1 est mise à 0, quel que soit l'état de signal à l'entrée E 0.4.

Si l'entrée E 0.0 ET l'entrée E 0.1 sont à 1, la sortie A 4.0 est mise à 1 si l'état de signal est 1 à l'entrée E 0.3.

Si l'entrée E 0.0 ET l'entrée E 0.1 sont à 0, la sortie A 4.0 reste inchangée, quel que soit l'état de signal à l'entrée E 0.3.

10.12 ---(MCRA) Activer relais de masquage

Remarques importantes sur l'utilisation de la fonctionnalité MCR

Représentation

---(MCRA)

Description de l'opération

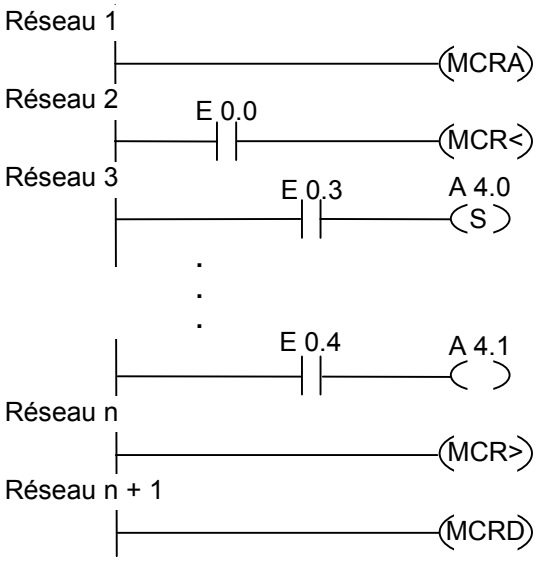
---(MCRA) (Activer relais de masquage)

Cette opération active la fonction de relais de masquage. (MCR : Master Control Relay). Vous pouvez, après cette commande, programmer des zones MCR en faisant appel aux opérations ---(MCR<) et ---(MCR>).

Mot d'état

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

Exemple



L'opération MCRA active la fonction MCR. Les opérations entre MCR< et MCR> (sorties A 4.0, A 4.1) sont exécutées comme suit :

Si l'entrée E 0.0 est à 1 (MCR **en fonction**), la sortie A 4.0 est mise à 1 si l'état de signal est 1 à l'entrée E 0.3 ou reste inchangée si l'état de signal est 0 à l'entrée E 0.3 et l'état de signal à l'entrée E 0.4 est affecté à la sortie A 4.1.

Si l'entrée E 0.0 est à 0 (MCR **hors fonction**), la sortie A 4.0 reste inchangée quel que soit l'état de signal à l'entrée E 0.3, et la sortie A 4.1 est mise à 0 quel que soit l'état de signal à l'entrée E 0.4.

L'opération ---(MCRD) désactive le relais MCR dans le trajet de courant suivant. Cela signifie que vous ne pouvez plus programmer de zones MCR avec ---(MCR<) et ---(MCR>).

10.13 ---(MCRD) Désactiver relais de masquage

Remarques importantes sur l'utilisation de la fonctionnalité MCR

Représentation

---(MCRD)

Description de l'opération

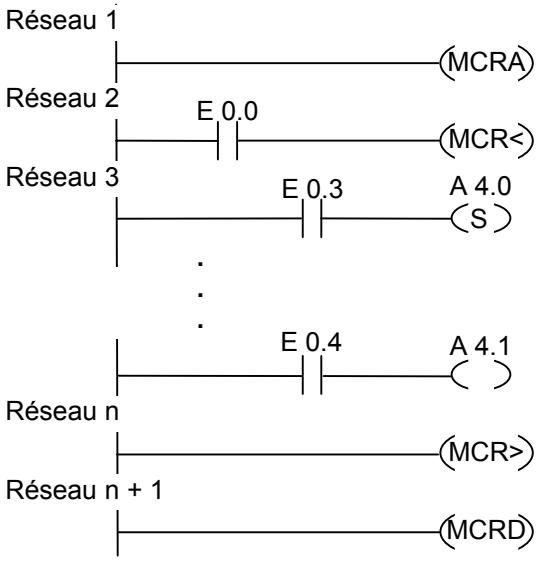
---(MCRD) (Désactiver relais de masquage)

Cette opération désactive la fonction de relais de masquage. Après cette opération, vous ne pouvez plus programmer de zones MCR.

Mot d'état

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

Exemple



L'opération MCRA active la fonction MCR. Les opérations entre MCR< et MCR> (sorties A 4.0, A 4.1) sont exécutées comme suit :

Si l'entrée E 0.0 est à 1 (MCR **en fonction**), la sortie A 4.0 est mise à 1 si l'état de signal est 1 à l'entrée E 0.3 ou reste inchangée si l'état de signal est 0 à l'entrée E 0.3 et l'état de signal à l'entrée E 0.4 est affecté à la sortie A 4.1.

Si l'entrée E 0.0 est à 0 (MCR **hors fonction**), la sortie A 4.0 reste inchangée quel que soit l'état de signal à l'entrée E 0.3, et la sortie A 4.1 est mise à 0 quel que soit l'état de signal à l'entrée E 0.4.

L'opération ---(MCRD) désactive le relais MCR dans le trajet de courant suivant. Cela signifie que vous ne pouvez plus programmer de zones MCR avec ---(MCR<) et ---(MCR>).

10.14 ---(RET) Retour

Représentation

---(RET)

Description de l'opération

RET (Retour)

Cette opération permet de quitter des blocs conditionnellement. Une combinaison amont est nécessaire pour cette sortie.

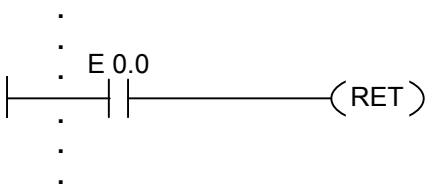
Mot d'état

Retour conditionnel (si RLG égale 1)

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

* L'opération **RET** 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). Le décalage vers la gauche multiplie le contenu de l'entrée IN par des puissances de 2 ; le décalage vers la droite le divise par des puissances de 2. 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.

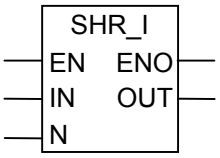
Le nombre de bits de décalage est précisé dans le paramètre d'entrée N. Les positions binaires libérées par l'opération de décalage sont soit remplies par des zéros, soit par l'état de signal du bit de signe (0 signifie positif et 1 négatif). L'état de signal du bit décalé en dernier est chargé dans 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



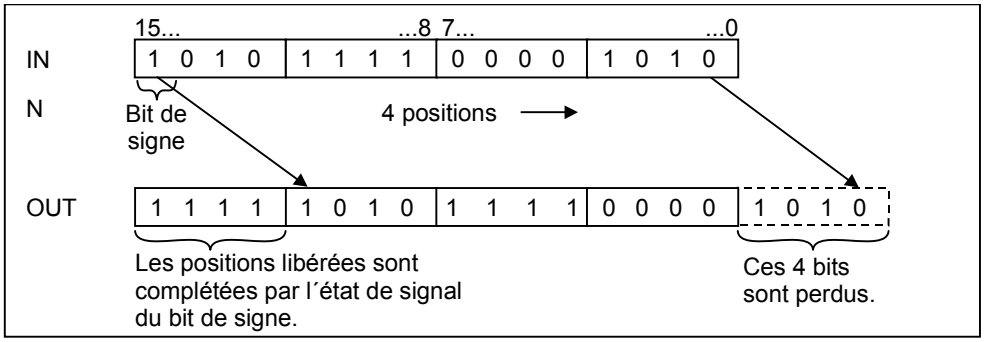
Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, L, D	Entrée de validation
ENO	BOOL	E, A, M, L, D	Sortie de validation
IN	INT	E, A, M, L, D	Valeur à décaler
N	WORD	E, A, M, L, D	Nombre de bits de décalage
OUT	INT	E, A, M, L, D	Résultat du décalage

Description de l'opération

SHR_I (Décalage vers la droite d'un entier de 16 bits)

Cette opération 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. Les bits 16 à 31 ne sont pas affectés. Le nombre de bits de décalage est indiqué dans l'entrée N. Si N est supérieur à 16, tout se passe comme si N était égal à 16. Les positions binaires libérées à gauche prennent l'état de signal du bit 15 (bit de signe du nombre entier). Elles prennent donc la valeur 0 s'il s'agit d'un nombre entier positif et la valeur 1 s'il s'agit d'un nombre entier négatif. Le résultat du décalage est rangé dans la sortie OUT. L'opération SHR_I met les bits BI0 et DEB à 0 si N est différent de 0.

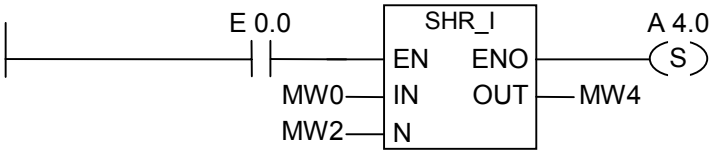
ENO a le même état de signal que EN.



Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	x	x	x	-	x	x	x	1

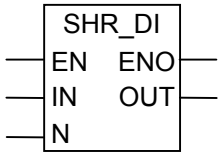
Exemple



L'opération SHR_I est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le mot de mémento MW0 est chargé et 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	Entrée de validation
ENO	BOOL	E, A, M, L, D	Sortie 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

Description de l'opération

SHR_DI (Décalage vers la droite d'un entier de 32 bits)

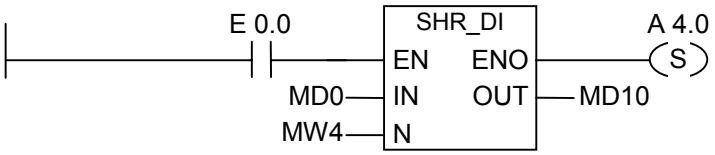
Cette opération est activée si l'état de signal est 1 à l'entrée de validation EN. Elle décale bit par bit vers la droite les bits 0 à 31 de l'entrée IN. Le nombre de bits de décalage est indiqué dans l'entrée N. Si N est supérieur à 32, tout se passe comme si N était égal à 32. Les positions binaires libérées à gauche prennent l'état de signal du bit 31 (bit de signe du nombre entier). Elles prennent donc la valeur 0 s'il s'agit d'un nombre entier positif et la valeur 1 s'il s'agit d'un nombre entier négatif. Le résultat du décalage est rangé dans la sortie OUT. L'opération SHR_DI met les bits BI0 et DEB à 0 si N est différent de 0.

ENO a le même état de signal que EN.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	x	x	x	-	x	x	x	1

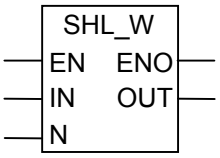
Exemple



L'opération SHR_DI est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le double mot de mémento MD0 est chargé et 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



Format

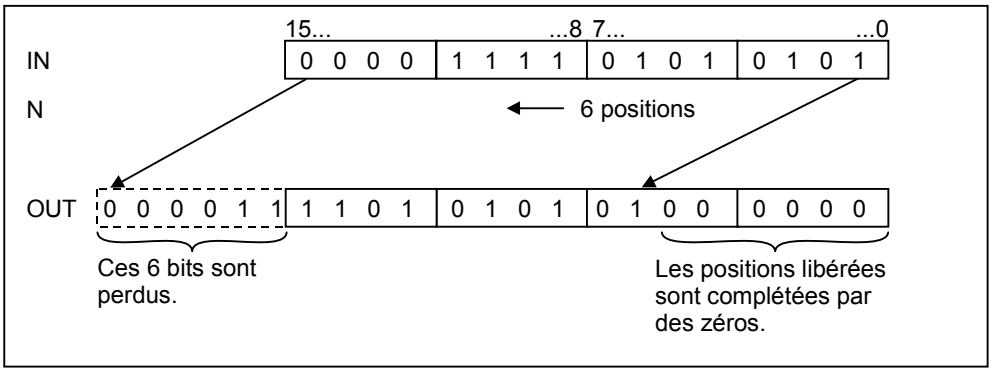
Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, L, D	Entrée de validation
ENO	BOOL	E, A, M, L, D	Sortie de validation
IN	WORD	E, A, M, L, D	Valeur à décaler
N	WORD	E, A, M, L, D	Nombre de bits de décalage
OUT	WORD	E, A, M, L, D	Résultat du décalage (mot)

Description de l'opération

SHL_W (Décalage vers la gauche d'un mot)

Cette opération 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. Les bits 16 à 31 ne sont pas influencés. Le nombre de bits de décalage est indiqué dans l'entrée N. Si N est supérieur à 16, la valeur 0 est écrite dans la sortie OUT et les bits BI0 et DEB du mot d'état sont mis à 0. Les N positions libérées à droite en raison du décalage sont complétées par des zéros. Le résultat du décalage est rangé dans la sortie OUT. L'opération SHL_W met les bits BI0 et DEB à 0 si N est différent de 0.

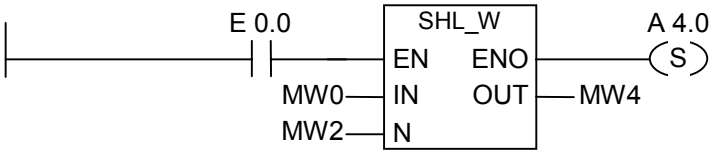
ENO a le même état de signal que EN.



Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	x	x	x	-	x	x	x	1

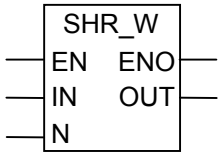
Exemple



L'opération SHL_W est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le mot de mémento MW0 est chargé et décalé vers la gauche du nombre de bits précisé dans MW2. Le résultat (mot) 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	Entrée de validation
ENO	BOOL	E, A, M, L, D	Sortie 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 (mot)

Description de l'opération

SHR_W (Décalage vers la droite d'un mot)

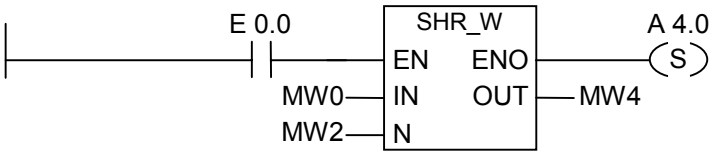
Cette opération est activée si l'état de signal est 1 à l'entrée de validation EN. Elle décale bit par bit vers la droite les bits 0 à 15 de l'entrée IN. Les bits 16 à 31 ne sont pas affectés. Le nombre de bits de décalage est indiqué dans l'entrée N. Si N est supérieur à 16, la valeur 0 est écrite dans la sortie OUT et les bits BI0 et DEB du mot d'état sont mis à 0. Les N positions libérées à gauche en raison du décalage sont complétées par des zéros. Le résultat du décalage est rangé dans la sortie OUT. L'opération SHR_W met les bits BI0 et DEB à 0 si N est différent de 0.

ENO a le même état de signal que EN.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	x	x	x	-	x	x	x	1

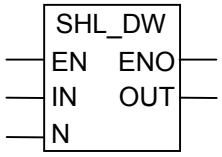
Exemple



L'opération SHR_W est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le mot de mémento MW0 est chargé et décalé vers la droite du nombre de bits précisé dans MW2. Le résultat (mot) 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	Entrée de validation
ENO	BOOL	E, A, M, L, D	Sortie 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 (double mot)

Description de l'opération

SHL_DW (Décalage vers la gauche d'un double mot)

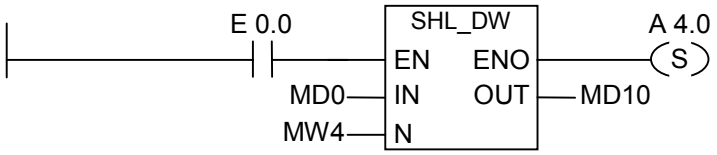
Cette opération est activée si l'état de signal est 1 à l'entrée de validation EN. Elle décale les bits 0 à 31 de l'entrée IN bit par bit vers la gauche. Le nombre de bits de décalage est indiqué dans l'entrée N. Si N est supérieur à 32, la valeur 0 est écrite dans la sortie OUT et les bits BI0 et DEB du mot d'état sont mis à 0. Les N positions libérées à droite en raison du décalage sont complétées par des zéros. Le résultat du décalage (double mot) est rangé dans la sortie OUT. L'opération SHL_DW met les bits BI0 et DEB à 0 si N est différent de 0.

ENO a le même état de signal que EN.

Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	x	x	x	-	x	x	x	1

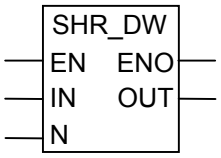
Exemple



L'opération SHL_DW est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le double mot de mémento MD0 est chargé et décalé vers la gauche du nombre de bits précisé dans MW4. Le résultat (double mot) 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



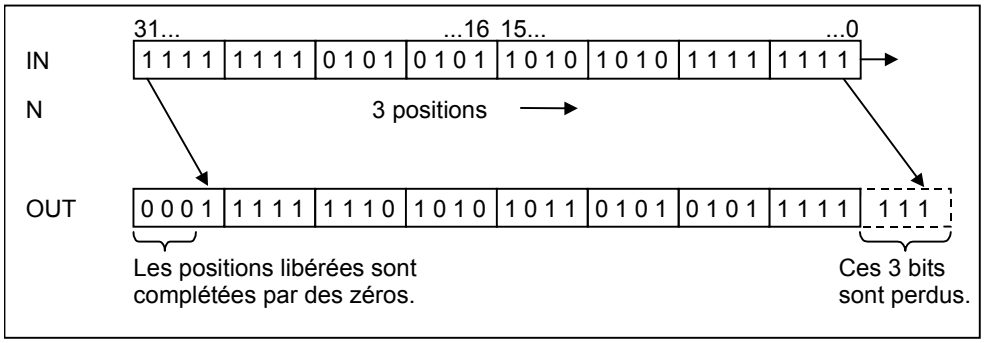
Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, L, D	Entrée de validation
ENO	BOOL	E, A, M, L, D	Sortie de validation
IN	DWORD	E, A, M, L, D	Valeur à décaler
N	WORD	E, A, M, L, D	Nombre de bits de décalage
OUT	DWORD	E, A, M, L, D	Résultat du décalage (double mot)

Description de l'opération

SHR_DW (Décalage vers la droite d'un double mot)

Cette opération 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 valeur 0 est écrite dans la sortie OUT et les bits BI0 et DEB du mot d'état sont mis à 0. Les N positions libérées à gauche en raison du décalage sont complétées par des zéros. Le résultat du décalage (double mot) est rangé dans la sortie OUT. L'opération SHR_DW met les bits BI0 et DEB à 0 si N est différent de 0.

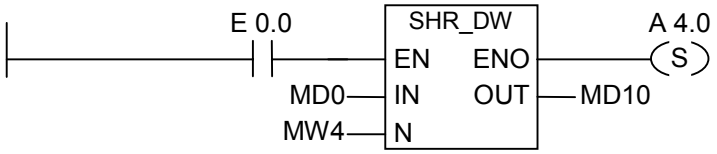
ENO a le même état de signal que EN.



Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	x	x	x	-	x	x	x	1

Exemple



L'opération SHR_DW est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le double mot de mémento MD0 est chargé et décalé vers la droite du nombre de bits précisé dans MW4. Le résultat (double mot) 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 (voir Registres de la CPU). 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.

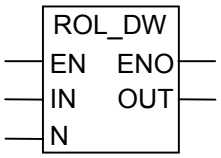
Selon l'opération, la rotation s'effectue via le bit BI1 du mot d'état. Le bit BI0 du mot d'état est remis à 0.

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



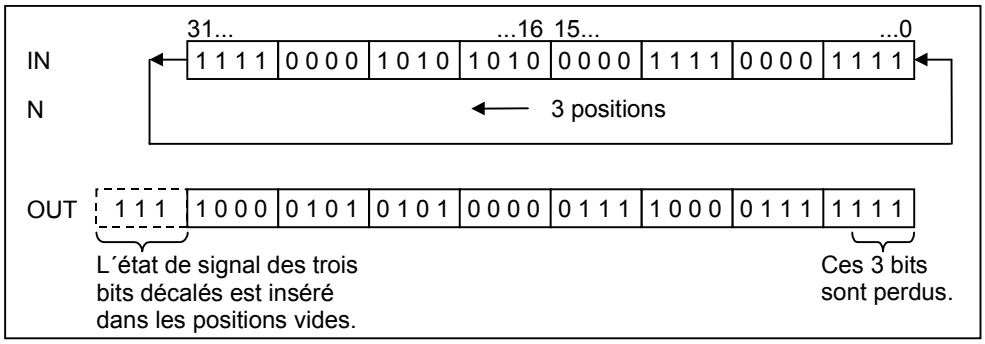
Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, L, D	Entrée de validation
ENO	BOOL	E, A, M, L, D	Sortie de validation
IN	DWORD	E, A, M, L, D	Valeur objet de la rotation
N	WORD	E, A, M, L, D	Nombre de bits de rotation
OUT	DWORD	E, A, M, L, D	Résultat de la rotation

Description de l'opération

ROL_DW (Rotation vers la gauche d'un double mot)

Cette opération 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 IN fait l'objet d'une rotation de ((N-1) modulo 32)+1 positions. Les positions binaires libérées à droite prennent l'état de signal des bits qui ont fait l'objet de la rotation vers la gauche. Le résultat de la rotation est rangé dans la sortie OUT. L'opération ROL_DW met les bits BI0 et DEB à 0 si N est différent de 0.

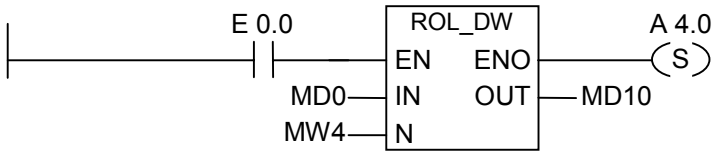
ENO a le même état de signal que EN.



Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	x	x	x	-	x	x	x	1

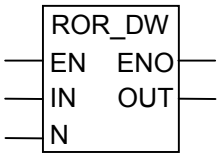
Exemple



L'opération ROL_DW est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le double mot de mémento MD0 est chargé et fait l'objet d'une rotation vers la gauche du nombre de bits précisé dans MW4. Le résultat (double mot) 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



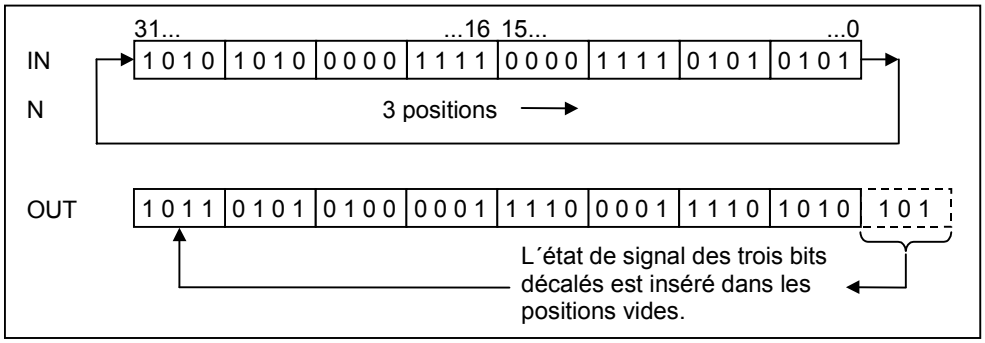
Paramètre	Type de données	Zone de mémoire	Description
EN	BOOL	E, A, M, L, D	Entrée de validation
ENO	BOOL	E, A, M, L, D	Sortie de validation
IN	DWORD	E, A, M, L, D	Valeur objet de la rotation
N	WORD	E, A, M, L, D	Nombre de bits de rotation
OUT	DWORD	E, A, M, L, D	Résultat de la rotation

Description de l'opération

ROR_DW (Rotation vers la droite d'un double mot)

Cette opération 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. Si N est supérieur à 32, le double mot IN fait l'objet d'une rotation de ((N-1) modulo 32)+1 positions. Les positions binaires libérées à gauche prennent l'état de signal des bits qui ont fait l'objet de la rotation vers la droite. Le résultat de la rotation est rangé dans la sortie OUT. L'opération ROR_DW met les bits BI0 et DEB à 0 si N est différent de 0.

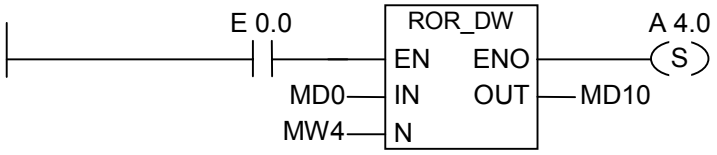
ENO a le même état de signal que EN.



Mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	x	x	x	-	x	x	x	1

Exemple



L'opération ROR_DW est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le double mot de mémento MD0 est chargé et fait l'objet d'une rotation vers la droite du nombre de bits précisé dans MW4. Le résultat (double mot) 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. Chacune de ces opérations réagit à l'une des conditions suivantes indiquées par un ou plusieurs bits du mot d'état :

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

Dans une connexion en série, les opérations sur bits d'état combinent le résultat de leur interrogation d'état de signal avec le résultat logique précédent selon la table de vérité ET. Dans une connexion en parallèle, elles combinent leur résultat avec le RLG précédent 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.

Structure du mot d'état :

2^{15}... ...2^{9}	2^{8}	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}
	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI

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 de l'opération

OV ---| |--- (Bit d'anomalie "débordement") et **OV ---| / |---** (Bit d'anomalie "débordement", forme inverse)

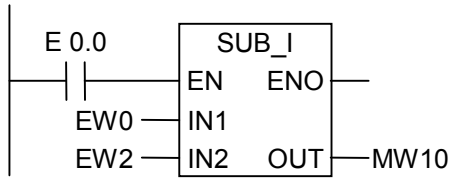
Ces opérations permettent de détecter un débordement dans l'opération arithmétique traitée en dernier (OV correspond à DEB). Cela signifie que le résultat de l'opération se situe hors de la plage positive ou négative autorisée. En cas de connexions en série, le résultat de l'interrogation est combiné au RLG par ET ; en cas de connexions en parallèle, il est combiné au RLG par OU.

Mot d'état

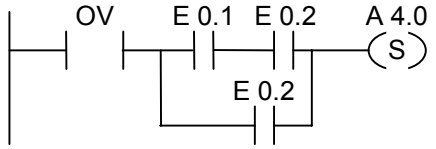
	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Ecriture	-	-	-	-	-	x	x	x	1

Exemple

Réseau 1



Réseau 2



L'état de signal 1 en E 0.0 active l'opération SUB_I. Si le résultat de l'opération arithmétique EW0 - EW2 est hors de la plage autorisée pour un nombre entier, le bit DEB est mis à 1.

Le résultat d'une interrogation d'état de signal en OV (DEB) égale 1.

Nota

L'interrogation de DEB n'est nécessaire qu'en raison de la présence de deux réseaux séparés. Sinon, si le résultat est hors de la plage autorisée, il suffit de considérer la sortie de validation ENO de l'opération arithmétique qui est à 0.

12.3 OS ---| |--- Bit d'anomalie "débordement mémorisé"

Représentation

Description de l'opération

OS ---| |--- (Bit d'anomalie "débordement mémorisé") et **OS ---| / |---** (Bit d'anomalie "débordement mémorisé", forme inverse)

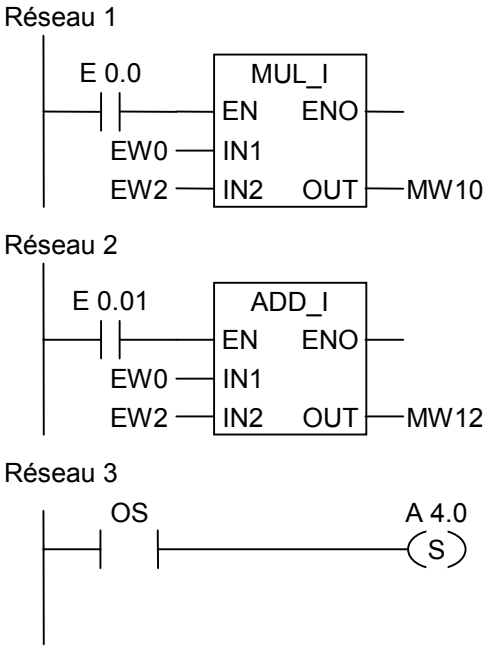
Ces opérations permettent de détecter et de mémoriser un débordement dans une opération arithmétique. Si le résultat de l'opération se situe hors de la plage positive ou négative autorisée, le bit DM (c'est-à-dire OS) du mot d'état est mis à 1. Contrairement au bit DEB qui est de nouveau écrit en cas d'opérations arithmétiques suivantes, le bit DM mémorise un débordement apparu. Le bit DM reste à 1 jusqu'à ce que le bloc soit quitté.

En cas de connexions en série, le résultat de l'interrogation est combiné au RLG par ET ; en cas de connexions en parallèle, il est combiné au RLG par OU.

Mot d'état

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

Exemple



L'opération MUL_I est activée par l'état de signal 1 en E 0.0 et l'opération ADD_I par l'état de signal 1 en E 0.1. Si le résultat de l'une des opérations arithmétiques est hors de la plage autorisée pour un nombre entier, le bit DM du mot d'état est mis à 1. La sortie A 4.0 est mise à 1 si l'interrogation de débordement mémorisé égale 1.

Nota

L'interrogation de DM n'est nécessaire qu'en raison de la présence de réseaux séparés. Sinon, il est possible de connecter la sortie ENO de la première opération arithmétique à l'entrée EN de la deuxième opération arithmétique (cascade).

12.4 UO ---| |--- Bit d'anomalie "illicite"

Représentation

Description de l'opération

UO ---| |--- (Bit d'anomalie "illicite") et **UO ---| / |---** (Bit d'anomalie "illicite", forme inverse)

Ces opérations permettent de détecter une opération arithmétique illicite sur nombres à virgule flottante, c'est-à-dire si l'une des valeurs dans l'opération arithmétique n'est pas un nombre à virgule flottante correct.

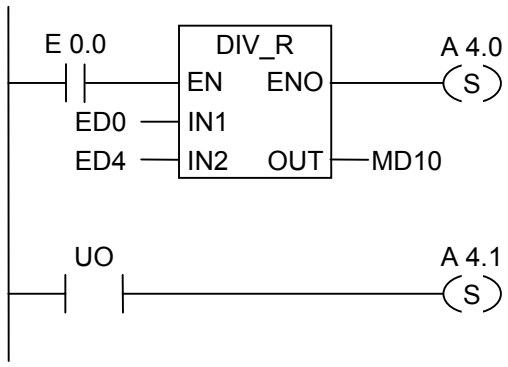
Si le résultat d´une opération arithmétique de nombres à virgule flottante est illicite (UO), l'interrogation d'état de signal égale 1. Si la combinaison dans BI1 et BI0 donne "non illicite", l'interrogation d'état de signal égale 0.

En cas de connexions en série, le résultat de l'interrogation est combiné au RLG par ET ; en cas de connexions en parallèle, il est combiné au RLG par OU.

Mot d'état

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

Exemple



L'état de signal 1 en E 0.0 active l'opération DIV_R. Si la valeur de ED0 ou ED4 ne correspond pas à un nombre à virgule flottante correct, l'opération arithmétique est illicite. Si l'état de signal en EN est 1 (activé) et qu'une erreur apparaisse pendant le traitement de la fonction DIV_R, l'état de signal en ENO sera égal à 0.

La sortie A 4.0 est mise à 1 si la fonction DIV_R est exécutée mais que l'une des valeurs n'est pas un nombre à virgule flottante correct.

12.5 BIE ---| |--- Bit d'anomalie "registre RB"

Représentation

Description de l'opération

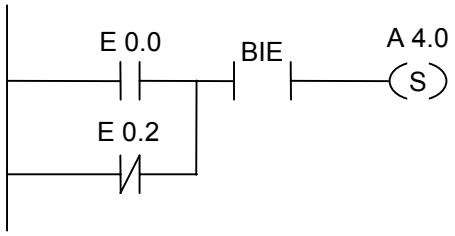
BIE ---| |--- (Bit d'anomalie registre RB) et **BIE ---| / |---** (Bit d'anomalie registre RB, forme inverse)

Ces opérations contrôlent l'état de signal du bit RB (correspond à BIE) dans le mot d'état. En cas de connexions en série, le résultat de l'interrogation est combiné au RLG par ET ; en cas de connexions en parallèle, il est combiné au RLG par OU. Le bit RB permet d'établir le lien entre traitement sur mots et traitement sur bits.

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'entrée E 0.0 égale 1 ou si l'entrée E 0.2 égale 0 et si, en plus de ce RLG, le bit RB égale 1.

12.6 ==0 ---| |--- Bit de résultat pour égal à 0

Représentation

Description de l'opération

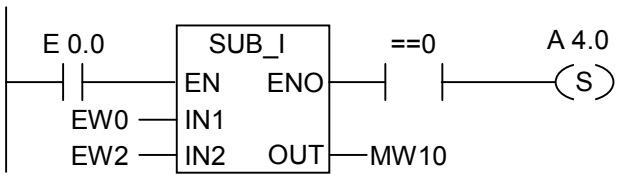
==0 ---| |--- (Bit de résultat pour égal à 0) et **==0 ---| / |---** (Bit de résultat pour égal à 0, forme inverse)

Ces opérations détectent si le résultat d´une opération arithmétique est égal à 0. Elles interrogent les bits indicateurs BI1 et BI0 afin de déterminer cette relation par rapport à 0. En cas de connexions en série, le résultat de l'interrogation est combiné au RLG par ET ; en cas de connexions en parallèle, il est combiné au RLG par OU.

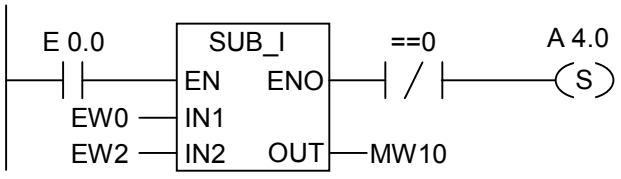
Mot d'état

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

Exemples



L'état de signal 1 en E 0.0 active cette opération. Si la valeur de EW0 est égale à la valeur de EW2, le résultat de l'opération arithmétique EW0 - EW2 est égal à 0. La sortie A 4.0 est mise à 1 si l'opération s'exécute sans erreur et si le résultat est égal à 0.



La sortie A 4.0 est mise à 1 si l'opération s'exécute sans erreur et si le résultat n'est pas égal à 0.

12.7 <>0 ---| |--- Bit de résultat pour différent de 0

Représentation

Description de l'opération

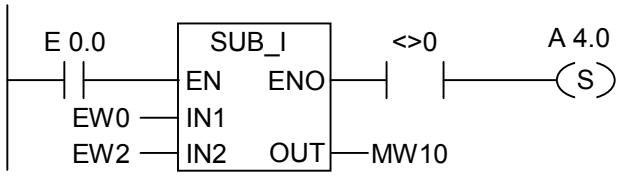
<>0 ---| |--- (Bit de résultat pour différent de 0) et **<>0 ---| / |---** (Bit de résultat pour différent de 0, forme inverse)

Ces opérations permettent de détecter si le résultat d´une opération arithmétique est différent de 0. Elles interrogent les bits indicateurs BI1 et BI0 afin de déterminer cette relation par rapport à 0. En cas de connexions en série, le résultat de l'interrogation est combiné au RLG par ET ; en cas de connexions en parallèle, il est combiné au RLG par OU.

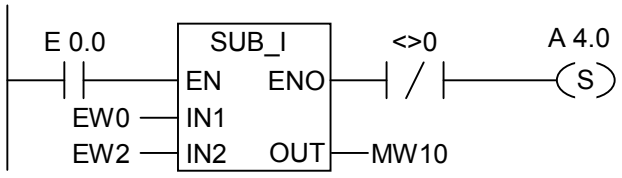
Mot d'état

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

Exemples



L'état de signal 1 en E 0.0 active cette opération. Si la valeur de EW0 est différente de la valeur de EW2, le résultat de l'opération arithmétique EW0 - EW2 est différent de 0. La sortie A 4.0 est mise à 1 si l'opération s'exécute sans erreur et si le résultat est différent de 0.



La sortie A 4.0 est mise à 1 si l'opération s'exécute sans erreur et si le résultat est égal à 0.

12.8 >=0 ---| |--- Bit de résultat pour supérieur ou égal à 0

Représentation

Description de l'opération

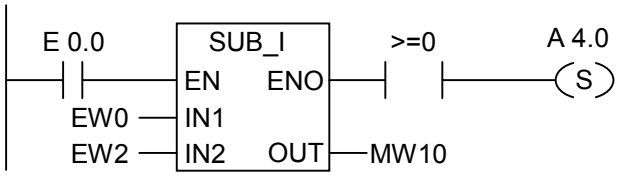
>=0 ---| |--- (Bit de résultat pour supérieur ou égal à 0) et **>=0 ---| / |---** (Bit de résultat pour supérieur ou égal à 0, forme inverse)

Ces opérations permettent de détecter si le résultat d'une opération arithmétique est supérieur ou égal à 0. Elles interrogent les bits indicateurs BI1 et BI0 afin de déterminer cette relation par rapport à 0. En cas de connexions en série, le résultat de l'interrogation est combiné au RLG par ET ; en cas de connexions en parallèle, il est combiné au RLG par OU.

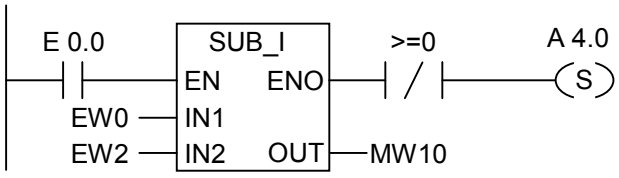
Mot d'état

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

Exemples



L'état de signal 1 en E 0.0 active cette opération. Si la valeur de EW0 est supérieure ou égale à la valeur de EW2, le résultat de l'opération arithmétique EW0 - EW2 est supérieur ou égal à 0. La sortie A 4.0 est mise à 1 si l'opération s'exécute sans erreur et si le résultat est supérieur ou égal à 0.



La sortie A 4.0 est mise à 1 si l'opération s'exécute sans erreur et si le résultat n'est pas supérieur ou égal à 0.

12.9 <=0 ---| |--- Bit de résultat pour inférieur ou égal à 0

Représentation

Description de l'opération

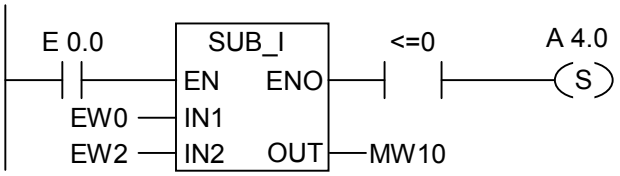
<=0 ---| |--- (Bit de résultat pour inférieur ou égal à 0) et **<=0 ---| / |---** (Bit de résultat pour inférieur ou égal à 0, forme inverse)

Ces opérations permettent de détecter si le résultat d´une opération arithmétique est inférieur ou égal à 0. Elles interrogent les bits indicateurs BI1 et BI0 afin de déterminer cette relation par rapport à 0. En cas de connexions en série, le résultat de l'interrogation est combiné au RLG par ET ; en cas de connexions en parallèle, il est combiné au RLG par OU.

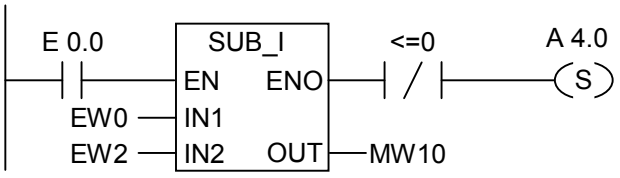
Mot d'état

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

Exemples



L'état de signal 1 en E 0.0 active cette opération. Si la valeur de EW0 est inférieure ou égale à la valeur de EW2, le résultat de l'opération arithmétique EW0 - EW2 est inférieur ou égal à 0. La sortie A 4.0 est mise à 1 si l'opération s'exécute sans erreur et si le résultat est inférieur ou égal à 0.



La sortie A 4.0 est mise à 1 si l'opération s'exécute sans erreur et si le résultat n'est pas inférieur ou égal à 0.

12.10 >0 ---| |--- Bit de résultat pour supérieur à 0

Représentation

Description de l'opération

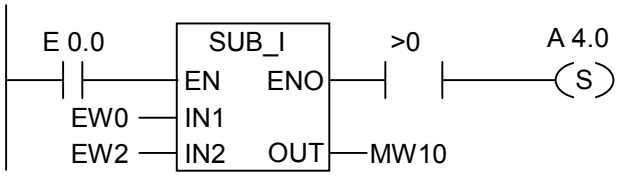
>0 ---| |--- (Bit de résultat pour supérieur à 0) et **>0 ---| / |---** (Bit de résultat pour supérieur à 0, forme inverse)

Ces opérations permettent de détecter si le résultat d'une opération arithmétique est supérieur à 0. Elles interrogent les bits indicateurs BI1 et BI0 afin de déterminer cette relation par rapport à 0. En cas de connexions en série, le résultat de l'interrogation est combiné au RLG par ET ; en cas de connexions en parallèle, il est combiné au RLG par OU.

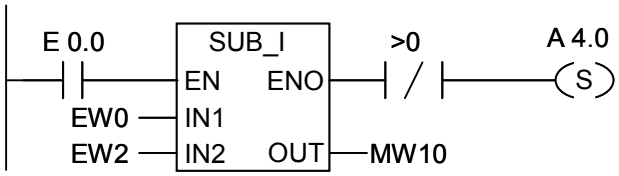
Mot d'état

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

Exemple



L'état de signal 1 en E 0.0 active cette opération. Si la valeur de EW0 est supérieure à la valeur de EW2, le résultat de l'opération arithmétique EW0 - EW2 est supérieur à 0. La sortie A 4.0 est mise à 1 si l'opération s'exécute sans erreur et si le résultat est supérieur à 0.



La sortie A 4.0 est mise à 1 si l'opération s'exécute sans erreur et si le résultat n'est pas supérieur à 0.

12.11 <0 ---| |--- Bit de résultat pour inférieur à 0

Représentation

Description de l'opération

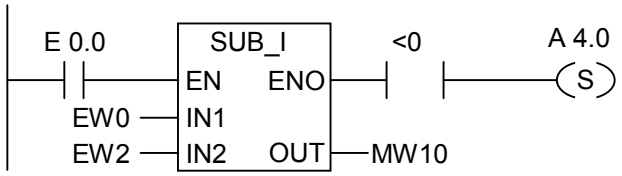
<0 ---| |--- (Bit de résultat pour inférieur à 0) et **<0 ---| / |---** (Bit de résultat pour inférieur à 0, forme inverse)

Ces opérations permettent de détecter si le résultat d'une opération arithmétique est inférieur à 0. Elles interrogent les bits indicateurs BI1 et BI0 afin de déterminer cette relation par rapport à 0. En cas de connexions en série, le résultat de l'interrogation est combiné au RLG par ET ; en cas de connexions en parallèle, il est combiné au RLG par OU.

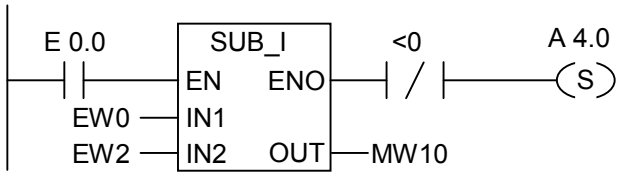
Mot d'état

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

Exemples



L'état de signal 1 en E 0.0 active cette opération. Si la valeur de EW0 est inférieure à la valeur de EW2, le résultat de l'opération arithmétique EW0 - EW2 est inférieur à 0. La sortie A 4.0 est mise à 1 si l'opération s'exécute sans erreur et si le résultat est inférieur à 0.



La sortie A 4.0 est mise à 1 si l'opération s'exécute sans erreur et si le résultat n'est pas inférieur à 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 temporisation sous forme d'impulsion
- S_VIMP Paramétrer et démarrer temporisation sous forme d'impulsion prolongée
- S_EVERZ Paramétrer et démarrer temporisation sous forme de retard à la montée
- S_SEVERZ Paramétrer et démarrer temporisation sous forme de retard à la montée mémorisé
- S_AVERZ Paramétrer et démarrer temporisation sous forme de retard à la retombée
- ---(SI) Démarrer temporisation sous forme d'impulsion
- ---(SV) Démarrer temporisation sous forme d'impulsion prolongée
- ---(SE) Démarrer temporisation sous forme de retard à la montée
- ---(SS) Démarrer temporisation sous forme de retard à la montée mémorisé
- ---(SA) Démarrer temporisation sous forme de retard à la retombée

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

Zone de mémoire

Une zone de mémoire est réservée aux temporisations dans votre CPU. Un mot de 16 bits y est réservé pour chaque opérande de temporisation. La programmation en CONT permet d'utiliser jusqu'à 256 temporisations. Le nombre de mots de temporisation disponibles dans votre CPU figure dans les caractéristiques de la CPU.

Les fonctions suivantes ont accès à la zone de mémoire réservée aux temporisations :

- opérations de temporisation,
- actualisation des mots de temporisation avec une horloge. Cette fonction décrémente, à l'état de 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 soit égale à zéro.

Valeur de temps

La valeur de temps est contenue sous forme binaire dans les bits 0 à 9 du mot de temporisation. Elle détermine un nombre d'unités. L'actualisation de l'heure décrémente la valeur de temps d'une unité dans un intervalle défini par la base de temps. La décrémentation se poursuit jusqu'à ce que la valeur de temps soit égale à zéro. Pour charger une valeur de temps, vous pouvez utiliser le format binaire, hexadécimal ou décimal codé binaire (DCB). La plage de temps est comprise entre 0 et 9 990 secondes.

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

- w#16#wxyz où
  - w = base de temps (c'est-à-dire l'intervalle de temps ou la résolution)
  - xyz = valeur de temps en format décimal codé binaire (DCB)
- 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

La base de temps est contenue en code binaire dans les bits 12 et 13 du mot de temporisation. Elle détermine à quel intervalle la valeur de temps va être décrémentée. La base de temps minimale est égale à 10 ms ; la base de temps maximale à 10 s.

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

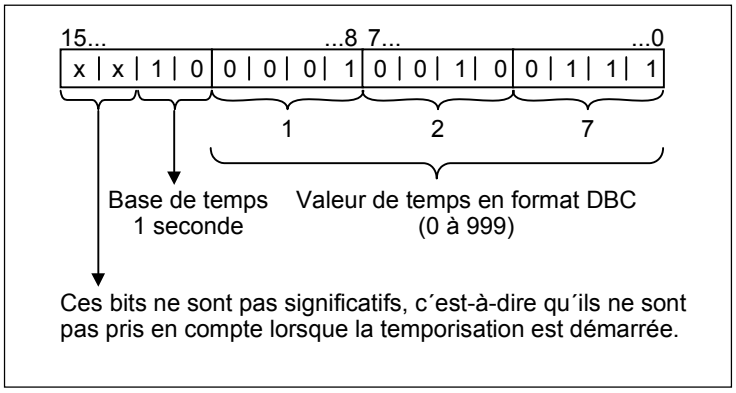
Les valeurs ne doivent pas excéder 2H_46M_30S. Les valeurs présentant une plage trop importante ou une trop grande résolution sont arrondies en fonction des valeurs limites de plages et de résolutions. Le format général S5TIME a les valeurs limites suivantes pour la plage et la résolution :

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 2H_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 et une base de temps de 1 seconde.

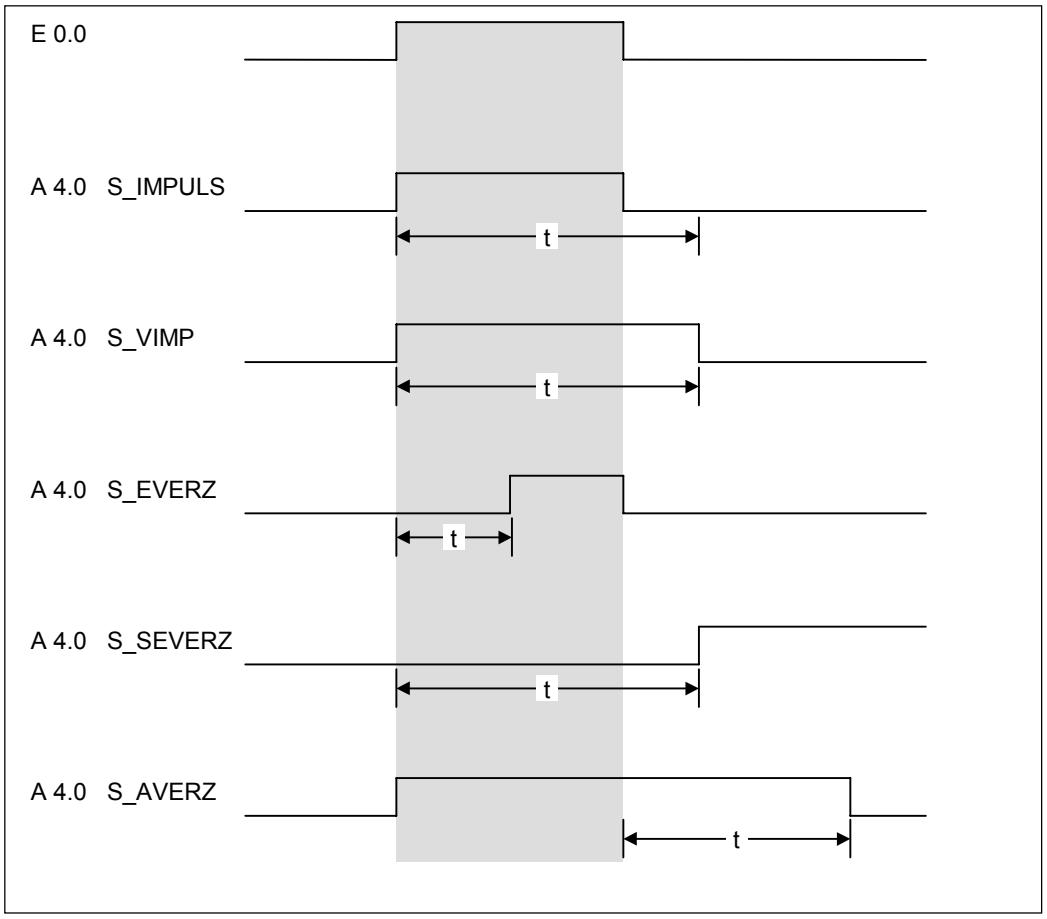


Lecture de la valeur et de la base de temps

Chaque boîte de temporisation possède deux sorties, DUAL (BI) et DEZ (BCD), pour lesquelles vous pouvez indiquer une adresse de mot. La sortie DUAL fournit la valeur de temps en format binaire. 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.

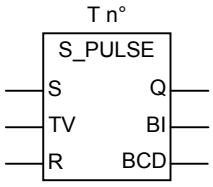


Temporisations	Description
S_IMPULS temporisation sous forme d´impulsion	La durée maximale pendant laquelle le signal de sortie reste à 1 est la même que la valeur de temps « t » programmée. Le signal de sortie reste à 1 pour une durée plus courte si le signal d'entrée passe à 0.
S_VIMP temporisation 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.

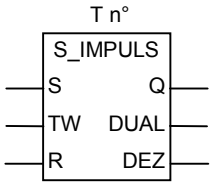
13.3 S_IMPULS Paramétrer et démarrer temporisation sous forme d'impulsion

Représentation

Anglaise



Allemande



Paramètre Anglaise	Paramètre Allemande	Type de données	Zone de mémoire	Description
T n°	T n°	TIMER	T	Numéro d'identification de la temporisation. La plage dépend de la CPU.
S	S	BOOL	E, A, M, L, D	Entrée de démarrage
TV	TW	S5TIME	E, A, M, L, D	Valeur de temps prédéfinie
R	R	BOOL	E, A, M, L, D	Entrée de remise à zéro
BI	DUAL	WORD	E, A, M, L, D	Valeur de temps restante (format binaire)
BCD	DEZ	WORD	E, A, M, L, D	Valeur de temps restante (format DCB)
Q	Q	BOOL	E, A, M, L, D	Etat de la temporisation

Description de l'opération

S_IMPULS (Paramétrer et démarrer temporisation sous forme d'impulsion)

Cette opération démarre la temporisation précisée en cas de front montant 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 égal à 1. Tant que la temporisation s'exécute, l'état de signal à la sortie Q égale 1. 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'état de signal à la sortie Q est 0.

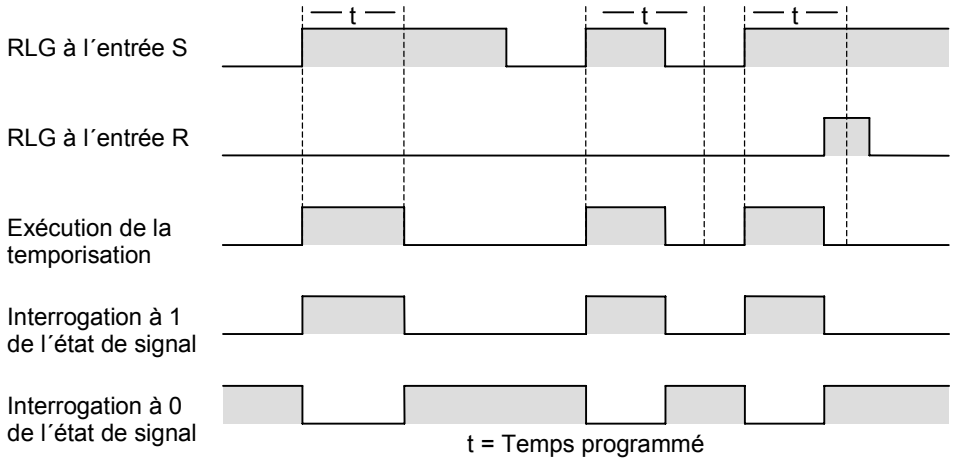
Si l'état de signal passe de 0 à 1 à l'entrée de remise à zéro R alors que la temporisation s'exécute, cette dernière est remise à zéro. La valeur de temps en cours et la base de temps sont alors également mises à 0. L'état de signal 1 à l'entrée R de la temporisation n'a aucun effet si la temporisation 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. La valeur de temps en cours correspond à la valeur initiale en TW moins la valeur de temps écoulée depuis le démarrage de la temporisation.

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

Chronogramme

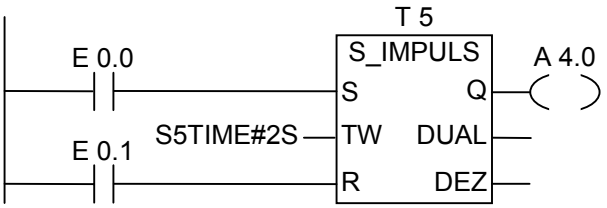
Propriétés de la temporisation sous forme d'impulsion



Mot d'état

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

Exemple



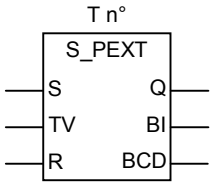
La temporisation T5 est démarrée si l'état de signal passe de 0 à 1 à l'entrée E 0.0 (front montant du RLG). Le temps de deux secondes (2 s) indiqué s'écoule tant que E 0.0 est à 1. Si l'état de signal en E 0.0 passe de 1 à 0 avant que le temps n'ait expiré, la temporisation s'arrête. Si l'état de signal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

L'état de signal à la sortie A 4.0 est 1 tant que la temporisation s'exécute. Cet état de signal est 0 si la temporisation a expiré ou si elle a été remise à zéro.

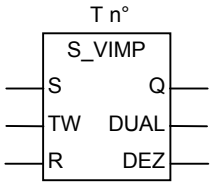
13.4 S_VIMP Paramétrer et démarrer temporisation sous forme d'impulsion prolongée

Représentation

Anglaise



Allemande



Paramètre Anglaise	Paramètre Allemande	Type de données	Zone de mémoire	Description
T n°	T n°	TIMER	T	Numéro d'identification de la temporisation. La plage dépend de la CPU.
S	S	BOOL	E, A, M, L, D	Entrée de démarrage
TV	TW	S5TIME	E, A, M, L, D	Valeur de temps prédéfinie
R	R	BOOL	E, A, M, L, D	Entrée de remise à zéro
BI	DUAL	WORD	E, A, M, L, D	Valeur de temps restante (format binaire)
BCD	DEZ	WORD	E, A, M, L, D	Valeur de temps restante (format DCB)
Q	Q	BOOL	E, A, M, L, D	Etat de la temporisation

Description de l'opération

S_VIMP (Paramétrer et démarrer temporisation sous forme d'impulsion prolongée)

Cette opération démarre la temporisation précisée en cas de front montant 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'exécute, l'état de signal à la sortie Q égale 1. La temporisation est redémarrée avec la valeur de temps prédéfinie si l'état de signal à l'entrée S passe de 0 à 1 alors que la temporisation s'exécute.

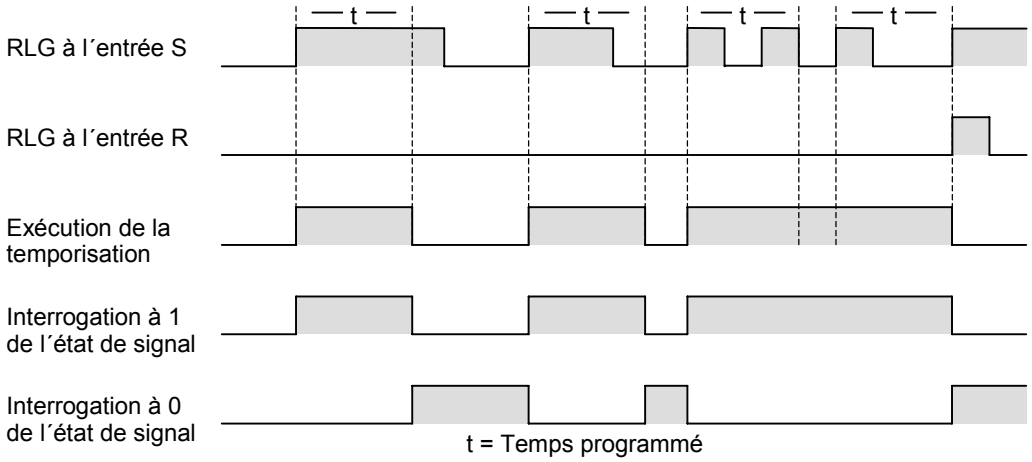
En cas de passage de 0 à 1 à l'entrée de remise à zéro R pendant que la temporisation s'exécute, cette dernière est remise à zéro. La valeur de temps en cours et la base de temps sont alors également mises à 0.

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. La valeur de temps en cours correspond à la valeur initiale en TW moins la valeur de temps écoulée depuis le démarrage de la temporisation.

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

Chronogramme

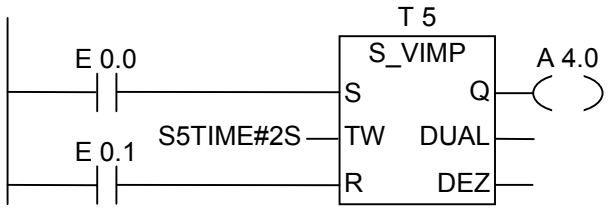
Propriétés de la temporisation sous forme d'impulsion prolongée



Mot d'état

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

Exemple

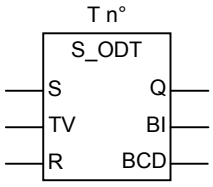


La temporisation T5 est démarrée si l'état de signal passe de 0 à 1 à l'entrée E 0.0 (front montant du RLG). Le temps de deux secondes (2 s) indiqué continue à s'écouler même en cas de front descendant à l'entrée S. Si l'état de signal en E 0.0 passe de 0 à 1 avant que le temps n'ait expiré, la temporisation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

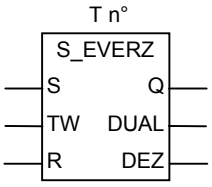
13.5 S_EVERZ Paramétrer et démarrer temporisation sous forme de retard à la montée

Représentation

Anglaise



Allemande



Paramètre Anglaise	Paramètre Allemande	Type de données	Zone de mémoire	Description
T n°	T n°	TIMER	T	Numéro d'identification de la temporisation. La plage dépend de la CPU.
S	S	BOOL	E, A, M, L, D	Entrée de démarrage
TV	TW	S5TIME	E, A, M, L, D	Valeur de temps prédéfinie
R	R	BOOL	E, A, M, L, D	Entrée de remise à zéro
BI	DUAL	WORD	E, A, M, L, D	Valeur de temps restante (format binaire)
BCD	DEZ	WORD	E, A, M, L, D	Valeur de temps restante (format DCB)
Q	Q	BOOL	E, A, M, L, D	Etat de la temporisation

Description de l'opération

S_EVERZ (Paramétrer et démarrer temporisation sous forme de retard à la montée)

Cette opération démarre la temporisation précisée en cas de front montant 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'état de signal à la sortie Q égale 1 lorsque la temporisation s'est exécutée sans erreur et que l'état de signal à l'entrée S est toujours 1. 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'état de signal à la sortie Q est 0.

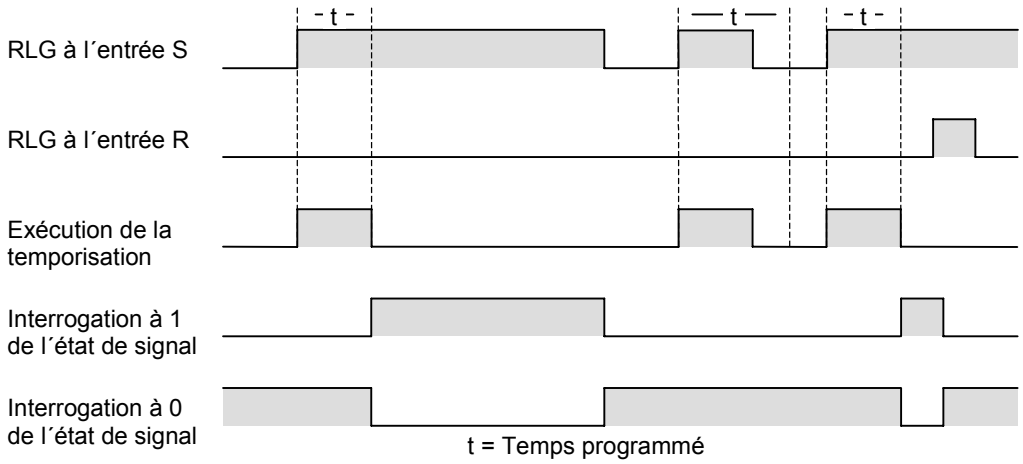
En cas de passage de 0 à 1 à l'entrée de remise à zéro R pendant que la temporisation s'exécute, cette dernière est remise à zéro. La valeur de temps en cours et la base de temps sont alors également mises à 0. L'état de signal à la sortie Q égale alors 0. La temporisation est également remise à zéro si l'état de signal égale 1 à l'entrée R alors que la temporisation ne s'exécute pas et que le RLG à l'entrée S est égal à 1.

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. La valeur de temps en cours correspond à la valeur initiale en TW moins la valeur de temps écoulée depuis le démarrage de la temporisation.

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

Chronogramme

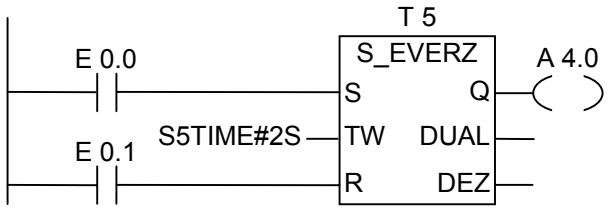
Propriétés de la temporisation sous forme de retard à la montée



Mot d'état

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

Exemple

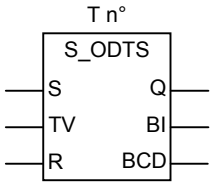


La temporisation T5 est démarrée si l'état de signal passe de 0 à 1 à l'entrée E 0.0 (front montant du RLG). Si le temps de deux secondes (2 s) indiqué expire et que l'état de signal à l'entrée E 0.0 égale toujours 1, l'état de signal à la sortie A 4.0 est 1. Si l'état de signal en E 0.0 passe de 1 à 0, la temporisation est arrêtée et A 4.0 est à 0. Si l'état de signal à l'entrée E 0.1 passe de 0 à 1, la temporisation est remise à zéro qu'elle soit en cours d'exécution ou non.

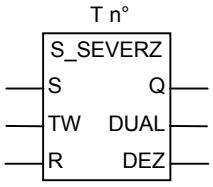
13.6 S_SEVERZ Paramétrer et démarrer temporisation sous forme de retard à la montée mémorisé

Représentation

Anglaise



Allemande



Paramètre Anglaise	Paramètre Allemande	Type de données	Zone de mémoire	Description
T n°	T n°	TIMER	T	Numéro d'identification de la temporisation. La plage dépend de la CPU.
S	S	BOOL	E, A, M, L, D	Entrée de démarrage
TV	TW	S5TIME	E, A, M, L, D	Valeur de temps prédéfinie
R	R	BOOL	E, A, M, L, D	Entrée de remise à zéro
BI	DUAL	WORD	E, A, M, L, D	Valeur de temps restante (format binaire)
BCD	DEZ	WORD	E, A, M, L, D	Valeur de temps restante (format DCB)
Q	Q	BOOL	E, A, M, L, D	Etat de la temporisation

Description de l'opération

S_SEVERZ (Paramétrer et démarrer temporisation sous forme de retard à la montée mémorisé)

Cette opération démarre la temporisation précisée en cas de front montant 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'état de signal à la sortie Q égale 1 lorsque la temporisation a expiré, quel que soit l'état de signal à l'entrée S. Si l'état de signal à l'entrée S passe de 0 à 1 alors que la temporisation s'exécute, cette dernière est redémarrée avec la valeur de temps indiquée.

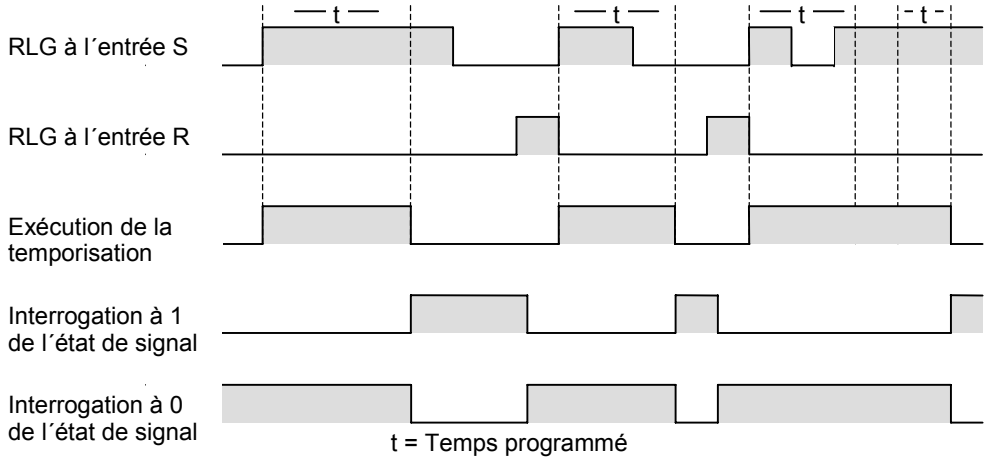
En cas de passage de 0 à 1 à l'entrée de remise à zéro R, la temporisation est remise à zéro quel que soit le RLG à l'entrée S. L'état de signal à la sortie Q est alors 0.

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. La valeur de temps en cours correspond à la valeur initiale en TW moins la valeur de temps écoulée depuis le démarrage de la temporisation.

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

Chronogramme

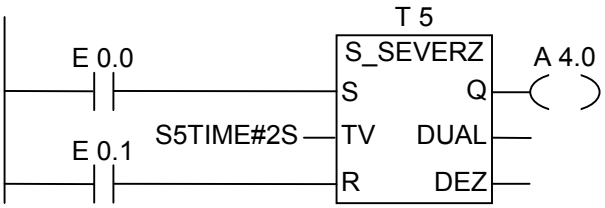
Propriétés de la temporisation sous forme de retard à la montée mémorisé



Mot d'état

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

Exemple

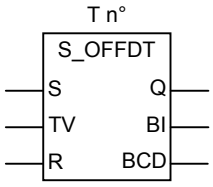


La temporisation T5 est démarrée si l'état de signal passe de 0 à 1 à l'entrée E 0.0 (front montant du RLG). La temporisation continue à s'exécuter même si l'état de signal en E 0.0 passe de 1 à 0. Si l'état de signal en E 0.0 passe de 0 à 1 avant que la temporisation n'ait expiré, la temporisation est redémarrée. L'état de signal à la sortie A 4.0 est 1 lorsque la temporisation s'est écoulée. Si l'état de signal à l'entrée E 0.1 passe de 0 à 1, la temporisation est remise à zéro quel que soit le RLG en S.

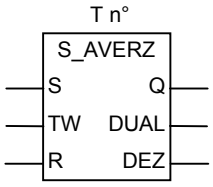
13.7 S_AVERZ Paramétrer et démarrer temporisation sous forme de retard à la retombée

Représentation

Anglaise



Allemande



Paramètre Anglaise	Paramètre Allemande	Type de données	Zone de mémoire	Description
T n°	T n°	TIMER	T	Numéro d'identification de la temporisation. La plage dépend de la CPU.
S	S	BOOL	E, A, M, L, D	Entrée de démarrage
TV	TW	S5TIME	E, A, M, L, D	Valeur de temps prédéfinie
R	R	BOOL	E, A, M, L, D	Entrée de remise à zéro
BI	DUAL	WORD	E, A, M, L, D	Valeur de temps restante (format binaire)
BCD	DEZ	WORD	E, A, M, L, D	Valeur de temps restante (format DCB)
Q	Q	BOOL	E, A, M, L, D	Etat de la temporisation

Description de l'opération

S_AVERZ (Paramétrer et démarrer temporisation sous forme de retard à la retombée)

Cette opération démarre la temporisation précisée en cas de front descendant à l'entrée de démarrage S. Un changement d'état de signal est toujours nécessaire pour activer une temporisation. L'état de signal à la sortie Q égale 1 lorsque l'état de signal à l'entrée S est 1 ou lorsque la temporisation s'exécute. 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.

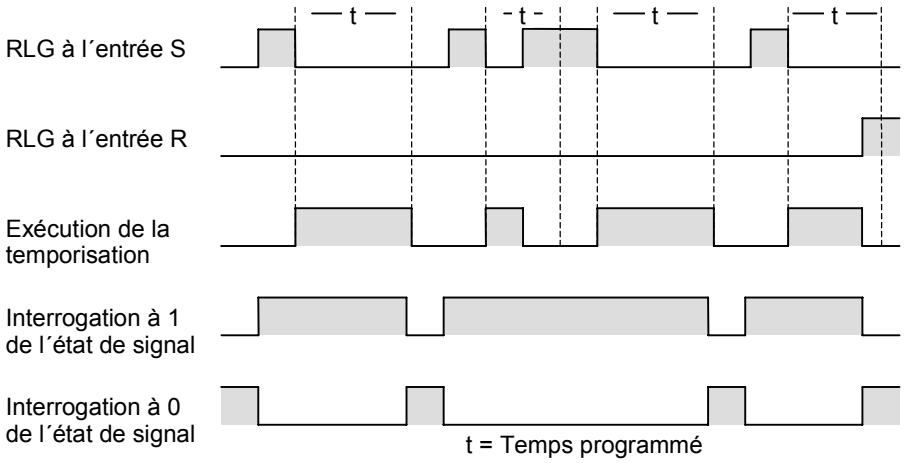
En cas de passage de 0 à 1 à l'entrée de remise à zéro R pendant que la temporisation s'exécute, cette dernière est remise à zéro.

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. La valeur de temps en cours correspond à la valeur initiale en TW moins la valeur de temps écoulée depuis le démarrage de la temporisation.

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

Chronogramme

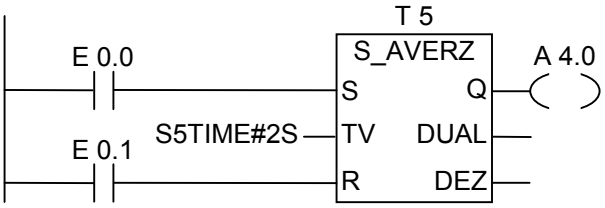
Propriétés de la temporisation sous forme de retard à la retombée



Mot d'état

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

Exemple



La temporisation 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 à 1 lorsque l'état de signal en E 0.0 est 1 ou que la temporisation s'exécute. Si l'état de signal en E 0.1 passe de 0 à 1 pendant que la temporisation s'exécute, cette dernière est remise à zéro.

13.8 ---(SI) Démarrer temporisation sous forme d'impulsion

Représentation

Anglaise	Allemande
<T n° >	<T n° >
---(SP)	**---(SI)**
<valeur de temps>	<valeur de temps>

Paramètre	Type de données	Zone de mémoire	Description
<T n° >	TIMER	T	Numéro de la temporisation. La plage dépend de la CPU.
<valeur de temps>	S5TIME	E, A, M, L, D	Valeur de temps prédéfinie

Description de l'opération

---(SI) (Démarrer temporisation sous forme d'impulsion)

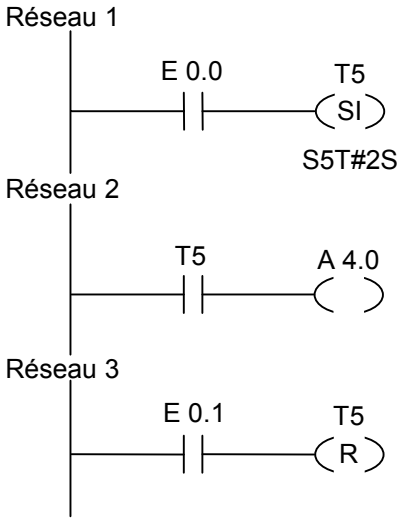
Cette opération démarre la temporisation indiquée avec la **<valeur de temps>** donnée si le RLG présente un front montant. La valeur de temps précisée continue à s'écouler tant que le RLG est positif (état de signal 1). L'interrogation à 1 de l'état de signal fournit un résultat égal à 1 tant que la temporisation s'exécute. Si le RLG passe de 1 à 0 avant que le temps indiqué ne soit écoulé, la temporisation s'arrête. Dans ce cas, l'interrogation à 1 de l'état de signal fournit un résultat égal à 0.

Voir aussi Adresse d´une temporisation en mémoire et composants d´une temporisation et S_IMPULS (Démarrer temporisation sous forme d'impulsion).

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 temporisation T5 est démarrée. La temporisation continue à s'exécuter 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. Si l'état de signal en E 0.1 passe de 0 à 1, la temporisation T5 est mise à zéro, c'est-à-dire qu'elle s'arrête et que la valeur de temps restante est mise à 0.

13.9 ---(SV) Démarrer temporisation sous forme d'impulsion prolongée

Représentation

Anglaise	Allemande
<T n° >	<T n° >
---(SE)	**---(SV)**
<valeur de temps>	<valeur de temps>

Paramètre	Type de données	Zone de mémoire	Description
<T n° >	TIMER	T	Numéro de la temporisation. La plage dépend de la CPU.
<valeur de temps>	S5TIME	E, A, M, L, D	Valeur de temps prédéfinie

Description de l'opération

---(SV) (Démarrer temporisation sous forme d'impulsion prolongée)

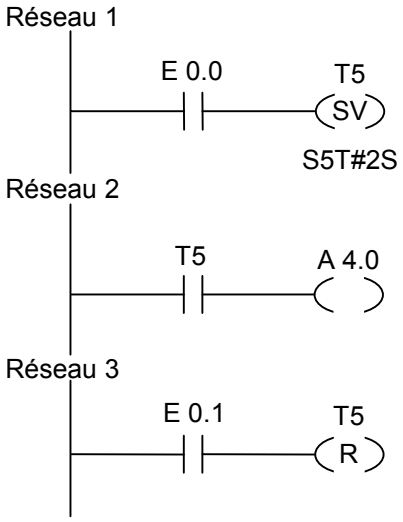
Cette opération démarre la temporisation indiquée avec la **<valeur de temps>** donnée si le RLG présente un front montant. La temporisation continue à s'exécuter avec la valeur de temps précisée même si le RLG passe à 0 avant que ce temps n'ait expiré. L'interrogation à 1 de l'état de signal fournit un résultat égal à 1 tant que la temporisation s'exécute. La temporisation est redémarrée avec la valeur de temps indiquée si le RLG passe de 0 à 1 alors que la temporisation s'exécute.

Voir aussi Adresse d´une temporisation en mémoire et composants d´une temporisation et S_VIMP (Démarrer temporisation sous forme d'impulsion prolongée).

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 temporisation T5 est démarrée. La temporisation continue à s'exécuter avec la valeur de temps précisée, même en présence d'un front descendant du RLG. Si l'état de signal en E 0.0 passe 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. Si l'état de signal en E 0.1 passe de 0 à 1, la temporisation T5 est mise à zéro, c'est-à-dire qu'elle s'arrête et que la valeur de temps restante est mise à 0.

13.10 ---(SE) Démarrer temporisation sous forme de retard à la montée

Représentation

Anglaise	Allemande
<T n° >	<T n° >
---(SD)	**---(SE)**
<valeur de temps>	<valeur de temps>

Paramètre	Type de données	Zone de mémoire	Description
<T n° >	TIMER	T	Numéro de la temporisation. La plage dépend de la CPU.
<valeur de temps>	S5TIME	E, A, M, L, D	Valeur de temps prédédinie

Description de l'opération

---(SE) (Démarrer temporisation sous forme de retard à la montée)

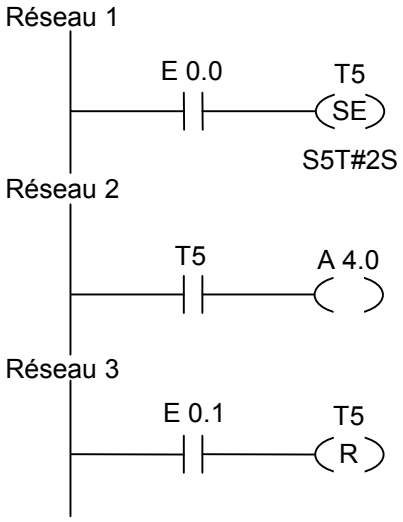
Cette opération démarre la temporisation indiquée avec la **<valeur de temps>** donnée si le RLG présente un front montant. L'interrogation à 1 de l'état de signal fournit un résultat égal à 1 lorsque la valeur de temps s'est écoulée sans erreur et que le RLG est toujours égal à 1. Si le RLG passe de 1 à 0 alors que la temporisation s'exécute, la temporisation est remise à zéro. Dans ce cas, l'interrogation à 1 de l'état de signal fournit un résultat égal à 0.

Voir aussi Adresse d´une temporisation en mémoire et composants d´une temporisation et S_EVERZ (Démarrer temporisation sous forme de retard à la montée).

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 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 à 1. Si l'état de signal en E 0.0 passe de 1 à 0 alors que la temporisation s'exécute, cette dernière est remise à zéro et A 4.0 est à 0. Si l'état de signal en E 0.1 passe de 0 à 1, la temporisation T5 est mise à zéro, c'est-à-dire qu'elle est arrêtée et que la valeur de temps restante est mise à 0.

13.11 ---(SS) Démarrer temporisation sous forme de retard à la montée mémorisé

Représentation

Anglaise	Allemande
<T n° >	<T n° >
---(SS)	**---(SS)**
<valeur de temps>	<valeur de temps>

Paramètre	Type de données	Zone de mémoire	Description
<T n° >	TIMER	T	Numéro de la temporisation. La plage dépend de la CPU.
<valeur de temps>	S5TIME	E, A, M, L, D	Valeur de temps prédéfinie

Description de l'opération

---(SS) (Démarrer temporisation sous forme de retard à la montée mémorisé)

Cette opération démarre la temporisation précisée si le RLG présente un front montant. L'état de signal de la temporisation est égal à 1 lorsque le temps indiqué est écoulé. Un redémarrage de la temporisation n'est possible que si cette dernière a été explicitement mise à zéro, car c'est uniquement de cette manière que l'état de signal de la temporisation peut être mis à 0.

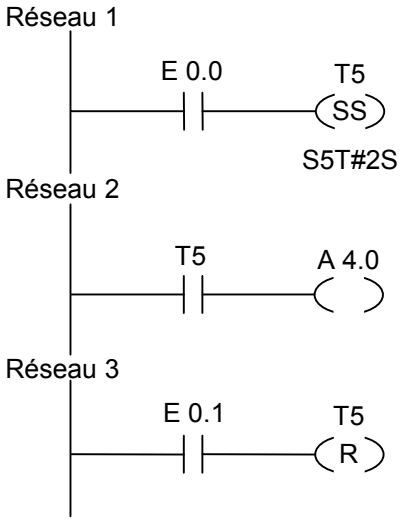
Si le RLG passe de 0 à 1 alors que la temporisation s'exécute, la temporisation est redémarrée avec le temps indiqué.

Voir aussi Adresse d´une temporisation en mémoire et composants d´une temporisation et S_SEVERZS (Démarrer temporisation sous forme de retard à la montée mémorisé).

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 temporisation T5 est démarrée. 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 1 si le temps a expiré. Si l'état de signal en E 0.1 est 1, la temporisation T5 est mise à zéro, c'est-à-dire qu'elle est arrêtée et que la valeur de temps restante est mise à 0.

13.12 ---(SA) Démarrer temporisation sous forme de retard à la retombée

Représentation

Anglaise	Allemande
<T n° >	<T n° >
---(SF)	**---(SA)**
<valeur de temps>	<valeur de temps>

Paramètre	Type de données	Zone de mémoire	Description
<T n° >	TIMER	T	Numéro de la temporisation. La plage dépend de la CPU.
<valeur de temps>	S5TIME	E, A, M, L, D	Valeur de temps prédéfinie

Description de l'opération

---(SA) (Démarrer temporisation sous forme de retard à la retombée)

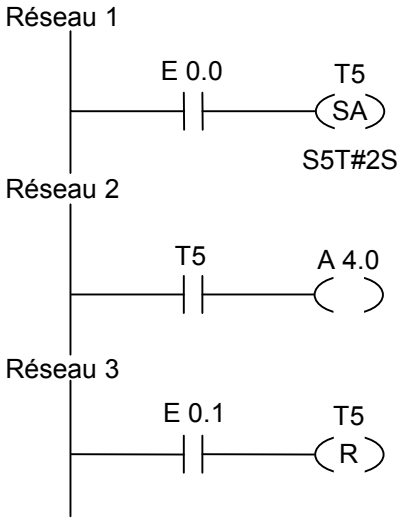
Cette opération démarre la temporisation indiquée si le RLG présente un front descendant. L'interrogation à 1 de l'état de signal fournit un résultat égal à 1 tant que le RLG égale 1 ou que la temporisation s'exécute avec la **<valeur de temps>** précisée. La temporisation est remise à zéro si le RLG passe de 0 à 1 alors que la temporisation s'exécute. La temporisation est toujours redémarrée lorsque le RLG repasse de 1 à 0.

Voir aussi Adresse d´une temporisation en mémoire et composants d´une temporisation et S_AVERZ (Démarrer temporisation sous forme de retard à la retombée).

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 1 à 0, la temporisation est démarrée.

L'état de signal à la sortie A 4.0 est à 1 si l'état de signal est 1 à l'entrée E 0.0 ou si la temporisation s'exécute. Si l'état de signal en E 0.1 passe de 0 à 1, la temporisation T5 est mise à zéro, c'est-à-dire qu'elle est arrêtée et que la valeur de temps restante est mise à 0.

14 Opérations combinatoires sur mots

14.1 Vue d'ensemble des opérations combinatoire sur mots

Description

Les opérations combinatoires sur mots combinent deux mots (16 bits) ou deux doubles mots (32 bits), bit par bit, selon les combinaisons booléennes. Ces opérations sont activées si l'état de signal est 1 à l'entrée de validation EN

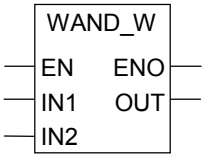
Si le résultat à la sortie OUT est différent de 0, le bit BI1 du mot d'état est mis à 1.
Si le résultat à la sortie OUT égale 0, le bit BI1 du mot d'état est mis à 0.

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, L, D	Entrée de validation
ENO	BOOL	E, A, M, L, D	Sortie de validation
IN1	WORD	E, A, M, L, D	Première valeur pour la combinaison
IN2	WORD	E, A, M, L, D	Seconde valeur pour la combinaison
OUT	WORD	E, A, M, L, D	Résultat de la combinaison (mot)

Description de l'opération

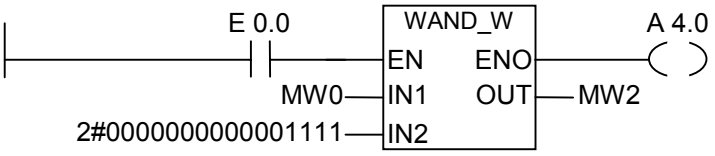
WAND_W (ET mot)

Cette opération est activée si l'état de signal est 1 à l'entrée de validation EN. Elle 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



Cette opération est exécutée si l'état de signal est 1 à l'entrée E 0.0. Seuls les bits 0 à 3 de MW0 sont significatifs ; les autres bits sont masqués par le profil binaire donné dans l'entrée IN2.

MW0 = 01010101 01010101

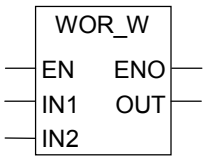
IN2 = 00000000 00001111

MW0 ET IN2 = MW2 = 00000000 00000101

La sortie A 4.0 est mise à 1 si l'opération est exécutée.

14.3 WOR_W OU mot

Représentation



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
IN1	WORD	E, A, M, L, D	Première valeur pour la combinaison
IN2	WORD	E, A, M, L, D	Seconde valeur pour la combinaison
OUT	WORD	E, A, M, L, D	Résultat de la combinaison (mot)

Description de l'opération

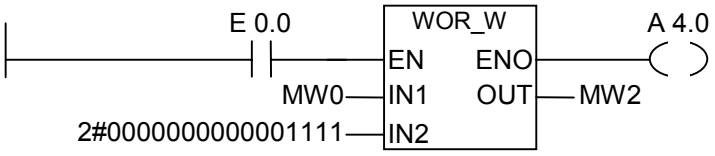
WOR_W (OU mot)

Cette opération est activée si l'état de signal est 1 à l'entrée de validation EN. Elle 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



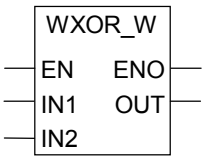
Cette opération est exécutée si l'état de signal est 1 à l'entrée E 0.0. Les bits 0 à 3 sont mis à 1, tous les autres bits de MW0 restent inchangés.

MW0	=	01010101 01010101
IN2	=	00000000 00001111
MW0 OU IN2=MW2	=	01010101 01011111

La sortie A 4.0 est mise à 1 si l'opération est exécutée.

14.4 WXOR_W OU exclusif mot

Représentation



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
IN1	WORD	E, A, M, L, D	Première valeur pour la combinaison
IN2	WORD	E, A, M, L, D	Seconde valeur pour la combinaison
OUT	WORD	E, A, M, L, D	Résultat de la combinaison (mot)

Description de l'opération

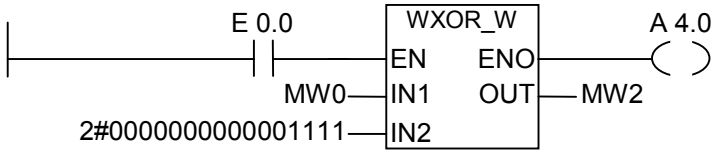
WXOR_W (OU exclusif mot)

Cette opération est activée si l'état de signal est 1 à l'entrée de validation EN. Elle 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



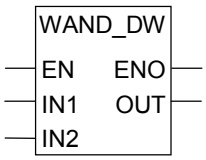
Cette opération est exécutée si l'état de signal est 1 à l'entrée E 0.0 :

MW0	=			01010101 01010101
IN2	=			00000000 00001111
MW0 OU exclusif IN2	=	MW2	=	01010101 01011010

La sortie A 4.0 est mise à 1 si l'opération est exécutée.

14.5 WAND_DW ET double mot

Représentation



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
IN1	DWORD	E, A, M, L, D	Première valeur pour la combinaison
IN2	DWORD	E, A, M, L, D	Seconde valeur pour la combinaison
OUT	DWORD	E, A, M, L, D	Résultat de la combinaison (double mot)

Description de l'opération

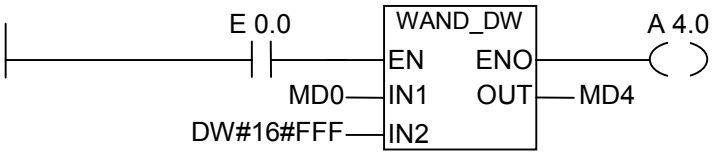
WAND_DW (ET double mot)

Cette opération est activée si l'état de signal est 1 à l'entrée de validation EN. Elle 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



Cette opération est exécutée si l'état de signal est 1 à l'entrée E 0.0. Seuls les bits 0 à 11 de MD0 sont significatifs ; les autres bits sont masqués par le profil binaire donné dans l'entrée IN2.

MD0 = 01010101 01010101 01010101 01010101

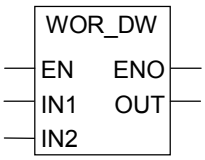
IN2 = 00000000 00000000 00001111 11111111

MD0 ET IN2 = MD4 = 00000000 00000000 00000101 01010101

La sortie A 4.0 est mise à 1 si l'opération est exécutée.

14.6 WOR_DW OU double mot

Représentation



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
IN1	DWORD	E, A, M, L, D	Première valeur pour la combinaison
IN2	DWORD	E, A, M, L, D	Seconde valeur pour la combinaison
OUT	DWORD	E, A, M, L, D	Résultat de la combinaison (double mot)

Description de l'opération

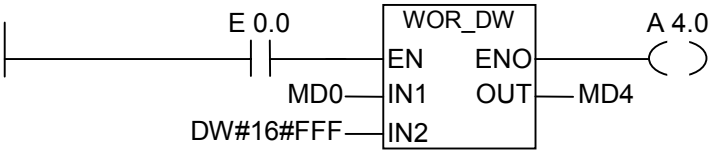
WOR_DW (OU double mot)

Cette opération est activée si l'état de signal est 1 à l'entrée de validation EN. Elle 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



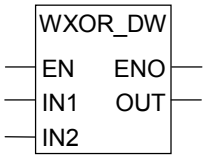
Cette opération est exécutée si l'état de signal est 1 à l'entrée E 0.0. Les bits 0 à 11 sont mis à 1. Les bits restants de MD0 restent inchangés.

MD0	=	01010101 01010101 01010101 01010101
IN2	=	00000000 00000000 00001111 11111111
MD0 OU IN2 = MD4	=	01010101 01010101 01011111 11111111

La sortie A 4.0 est mise à 1 si l'opération est exécutée.

14.7 WXOR_DW OU exclusif double mot

Représentation



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
IN1	DWORD	E, A, M, L, D	Première valeur pour la combinaison
IN2	DWORD	E, A, M, L, D	Seconde valeur pour la combinaison
OUT	DWORD	E, A, M, L, D	Résultat de la combinaison (double mot)

Description de l'opération

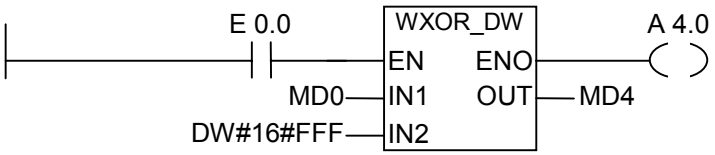
WXOR_DW (OU exclusif double mot)

Cette opération est activée si l'état de signal est 1 à l'entrée de validation EN. Elle 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



Cette opération est exécutée si l'état de signal est 1 à l'entrée E 0.0 :

MD0 = 01010101 01010101 01010101 01010101

IN2 = 00000000 00000000 00001111 11111111

MD0 OU exclusif IN2 = MD4 = 01010101 01010101 01011010 10101010

La sortie A 4.0 est mise à 1 si l'opération est exécutée.

A Présentation de toutes les opérations CONT

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

Abréviations allemandes	Abréviations anglaises	Catalogue des éléments de programme	Description
---| |---	---| |---	Combinaison sur bits	Contact à fermeture
---|/|---	---|/|---	Combinaison sur bits	Contact à ouverture
---( )	---( )	Combinaison sur bits	Bobine de sortie
---(#)---	---(#)---	Combinaison sur bits	Connecteur
==0 ---| |---	==0 ---| |---	Bits d'état	Bit de résultat pour égal à 0
>0 ---| |---	>0 ---| |---	Bits d'état	Bit de résultat pour supérieur à 0
>=0 ---| |---	>=0 ---| |---	Bits d'état	Bit de résultat pour supérieur ou égal à 0
<=0 ---| |---	<=0 ---| |---	Bits d'état	Bit de résultat pour inférieur ou égal à 0
<0 ---| |---	<0 ---| |---	Bits d'état	Bit de résultat pour inférieur à 0
<>0 ---| |---	<>0 ---| |---	Bits d'état	Bit de résultat pour différent de 0
ABS	ABS	Fonction sur nombres à virgule flottante	Valeur absolue d'un nombre réel
ACOS	ACOS	Fonction sur nombres à virgule flottante	Arc cosinus
ADD_DI	ADD_DI	Fonction sur nombres entiers	Additionner entiers de 32 bits
ADD_I	ADD_I	Fonction sur nombres entiers	Additionner entiers de 16 bits
ADD_R	ADD_R	Fonction sur nombres à virgule flottante	Additionner réels
ASIN	ASIN	Fonction sur nombres à virgule flottante	Arc sinus
ATAN	ATAN	Fonction sur nombres à virgule flottante	Arc tangente
---(OPN)	---(OPN)	Appel de DB	Ouvrir bloc de données
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	Appeler FB (boîte)
CALL_FC	CALL_FC	Gestion d'exécution de programmes	Appeler FC (boîte)
CALL_SFB	CALL_SFB	Gestion d'exécution de programmes	Appeler SFB (boîte)

Abréviations allemandes	Abréviations anglaises	Catalogue des éléments de programme	Description
CALL_SFC	CALL_SFC	Gestion d'exécution de programmes	Appeler SFC (boîte)
CEIL	CEIL	Conversion	Convertir 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 réels
COS	COS	Fonction sur nombres à virgule flottante	Cosinus
DI_BCD	DI_BCD	Conversion	Convertir entier de 32 bits en nombre DCB
DI_R	DI_R	Conversion	Convertir entier de 32 bits en 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 entiers	Diviser réels
EXP	EXP	Fonction sur nombres à virgule flottante	Valeur exponentielle
FLOOR	FLOOR	Conversion	Convertir 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 à l'intérieur d'un bloc si 1 (conditionnel)
---(JMPN)	---(JMPN)	Sauts	Saut à l'intérieur d'un bloc si 0 (conditionnel)
LABEL	LABEL	Sauts	Repère de saut
LN	LN	Fonction sur nombres à virgule flottante	Logarithme naturel
---(MCR>)	---(MCR>)	Gestion d'exécution de programmes	Relais de masquage hors fonction
---(MCR<)	---(MCR<)	Gestion d'exécution de programmes	Relais de masquage en fonction
---(MCRA)	---(MCRA)	Gestion d'exécution de programmes	Activer relais de masquage
---(MCRD)	---(MCRD)	Gestion d'exécution de programmes	Désactiver relais de masquage
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 réels

Abréviations allemandes	Abréviations anglaises	Catalogue des éléments de programme	Description
---(N)---	---(N)---	Combinaison sur bits	Détecter front descendant
NEG	NEG	Combinaison sur bits	Détecter front descendant de signal
NEG_DI	NEG_DI	Conversion	Complément à 2 d'entier de 32 bits
NEG_I	NEG_I	Conversion	Complément à 2 d'entier de 16 bits
NEG_R	NEG_R	Conversion	Inverser le signe d'un nombre réel
---| NOT |---	---| NOT |---	Combinaison sur bits	Inverser RLG
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
ROR_DW	ROR_DW	Décalage/rotation	Rotation vers la droite d'un double mot
ROUND	ROUND	Conversion	Arrondir
RS	RS	Combinaison sur bits	Bascule mise à 0, mise à 1
---(S)	---(S)	Combinaison sur bits	Mettre à 1
---(SA)	---(SF)	Temporisations	Démarrer 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 temporisation sous forme de retard à la retombée
---(SE)	---(SD)	Temporisations	Démarrer temporisation sous forme de retard à la montée
S_EVERZ	S_ODT	Temporisations	Paramétrer et démarrer 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	Démarrer temporisation sous forme d'impulsion
S_IMPULS	S_PULSE	Temporisations	Paramétrer et démarrer temporisation sous forme d'impulsion
SIN	SIN	Fonction sur nombres à virgule flottante	Sinus
SQR	SQR	Fonction sur nombres à virgule flottante	Carré
SQRT	SQRT	Fonction sur nombres à virgule flottante	Racine carrée
SR	SR	Combinaison sur bits	Bascule mise à 1, mise à 0
---(SS)	---(SS)	Temporisations	Démarrer temporisation sous forme de retard à la montée mémorisé

Abréviations allemandes	Abréviations anglaises	Catalogue des éléments de programme	Description
S_SEVERZ	S_ODTS	Temporisations	Paramétrer et démarrer temporisation sous forme de retard à la montée mémorisé (SS)
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 réels
---(SV)	---(SE)	Temporisations	Démarrer temporisation sous forme d'impulsion prolongée
S_VIMP	S_PEXT	Temporisations	Paramétrer et démarrer temporisation sous forme d'impulsion prolongée
---(SZ)	---(SC)	Compteurs	Initialiser compteur
TAN	TAN	Fonction sur nombres à virgule flottante	Tangente
TRUNC	TRUNC	Conversion	Tronquer à la partie entière
UO ---| |---	UO ---| |---	Bits d'état	Bit d'anomalie "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
ZAEHLER	S_CUD	Compteurs	Paramétrage et compteur d'incrémentation/décrémentation
----(ZR)	----(CD)	Compteurs	Décrémenter
Z_RUECK	----(S_CD)	Compteurs	Paramétrage et compteur de décrémentation
---(ZV)	----(CU)	Compteurs	Incrémenter
Z_VORW	S_CU	Compteurs	Paramétrage et compteur d'incrémentation

A.2 Opérations CONT classées d'après les abréviations anglaises (International)

Abréviations anglaises	Abréviations allemandes	Catalogue des éléments de programme	Description
---|/|---	---|/|---	Combinaison sur bits	Contact à ouverture
---| |---	---| |---	Combinaison sur bits	Contact à fermeture
---( )	---( )	Combinaison sur bits	Bobine de sortie
---(#)---	---(#)---	Combinaison sur bits	Connecteur
==0 ---| |---	==0 ---| |---	Bits d'état	Bit de résultat pour égal à 0
>0 ---| |---	>0 ---| |---	Bits d'état	Bit de résultat pour supérieur à 0
>=0 ---| |---	>=0 ---| |---	Bits d'état	Bit de résultat pour supérieur ou égal à 0
<=0 ---| |---	<=0 ---| |---	Bits d'état	Bit de résultat pour inférieur ou égal à 0
<0 ---| |---	<0 ---| |---	Bits d'état	Bit de résultat pour inférieur à 0
<>0 ---| |---	<>0 ---| |---	Bits d'état	Bit de résultat pour différent de 0
ABS	ABS	Fonction sur nombres à virgule flottante	Valeur absolue d'un nombre réel
ACOS	ACOS	Fonction sur nombres à virgule flottante	Arc cosinus
ADD_DI	ADD_DI	Fonction sur nombres entiers	Additionner entiers de 32 bits
ADD_I	ADD_I	Fonction sur nombres entiers	Additionner entiers de 16 bits
ADD_R	ADD_R	Fonction sur nombres à virgule flottante	Additionner réels
ASIN	ASIN	Fonction sur nombres à virgule flottante	Arc sinus
ATAN	ATAN	Fonction sur nombres à virgule flottante	Arc tangente
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	Appeler FB (boîte)
CALL_FC	CALL_FC	Gestion d'exécution de programmes	Appeler FC (boîte)
CALL_SFB	CALL_SFB	Gestion d'exécution de programmes	Appeler SFB (boîte)
CALL_SFC	CALL_SFC	Gestion d'exécution de programmes	Appeler SFC (boîte)
----(CD)	----(ZR)	Compteurs	Décrémenter
CEIL	CEIL	Conversion	Convertir 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

Abréviations anglaises	Abréviations allemandes	Catalogue des éléments de programme	Description
CMP ? R	CMP ? R	Comparaison	Comparer réels
COS	COS	Fonction sur nombres à virgule flottante	Cosinus
----(CU)	---(ZV)	Compteurs	Incrémenter
DI_BCD	DI_BCD	Conversion	Convertir entier de 32 bits en nombre DCB
DI_R	DI_R	Conversion	Convertir entier de 32 bits en 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 réels
EXP	EXP	Fonction sur nombres à virgule flottante	Valeur exponentielle
FLOOR	FLOOR	Conversion	Convertir 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 à l'intérieur d'un bloc si 1 (conditionnel)
---(JMPN)	---(JMPN)	Sauts	Saut à l'intérieur d'un bloc si 0 (conditionnel)
LABEL	LABEL	Sauts	Repère de saut
LN	LN	Fonction sur nombres à virgule flottante	Logarithme naturel
---(MCR>)	---(MCR>)	Gestion d'exécution de programmes	Relais de masquage hors fonction
---(MCR<)	---(MCR<)	Gestion d'exécution de programmes	Relais de masquage en fonction
---(MCRA)	---(MCRA)	Gestion d'exécution de programmes	Activer relais de masquage
---(MCRD)	---(MCRD)	Gestion d'exécution de programmes	Désactiver relais de masquage
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 réels
---(N)---	---(N)---	Combinaison sur bits	Détecter front descendant
NEG	NEG	Combinaison sur bits	Détecter front descendant de signal
NEG_DI	NEG_DI	Conversion	Complément à 2 d'entier de 32 bits
NEG_I	NEG_I	Conversion	Complément à 2 d'entier de 16 bits

Abréviations anglaises	Abréviations allemandes	Catalogue des éléments de programme	Description
NEG_R	NEG_R	Conversion	Inverser le signe d'un nombre réel
---| NOT |---	---| NOT |---	Combinaison sur bits	Inverser RLG
---(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
ROR_DW	ROR_DW	Décalage/rotation	Rotation vers la droite d'un double mot
ROUND	ROUND	Conversion	Arrondir
RS	RS	Combinaison sur bits	Bascule mise à 0, mise à 1
---(S)	---(S)	Combinaison sur bits	Mettre à 1
---(SAVE)	---(SAVE)	Combinaison sur bits	Sauvegarder RLG dans RB
---(SC)	---(SZ)	Compteurs	Initialiser compteur
----(S_CD)	Z_RUECK	Compteurs	Paramétrage et compteur de décrémentation
S_CU	Z_VORW	Compteurs	Paramétrage et compteur d'incrémentation
S_CUD	ZAEHLER	Compteurs	Paramétrage et compteur d'incrémentation/décrémentation
---(SD)	---(SE)	Temporisations	Démarrer temporisation sous forme de retard à la montée
---(SE)	---(SV)	Temporisations	Démarrer temporisation sous forme d'impulsion prolongée
---(SF)	---(SA)	Temporisations	Démarrer 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	Sinus
S_ODT	S_EVERZ	Temporisations	Paramétrer et démarrer temporisation sous forme de retard à la montée
S_ODTS	S_SEVERZ	Temporisations	Paramétrer et démarrer temporisation sous forme de retard à la montée mémorisé (SS)
S_OFFDT	S_AVERZ	Temporisations	Paramétrer et démarrer temporisation sous forme de retard à la retombée
---(SP)	---(SI)	Temporisations	Démarrer temporisation sous forme d'impulsion
S_PEXT	S_VIMP	Temporisations	Paramétrer et démarrer temporisation sous forme d'impulsion prolongée

Abréviations anglaises	Abréviations allemandes	Catalogue des éléments de programme	Description
S_PULSE	S_IMPULS	Temporisations	Paramétrer et démarrer temporisation sous forme d'impulsion
SQR	SQR	Fonction sur nombres à virgule flottante	Carré
SQRT	SQRT	Fonction sur nombres à virgule flottante	Racine carrée
SR	SR	Combinaison sur bits	Bascule mise à 1, mise à 0
---(SS)	---(SS)	Temporisation	Démarrer 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 réels
TAN	TAN	Fonction sur nombres à virgule flottante	Tangente
TRUNC	TRUNC	Conversion	Tronquer à la partie entière
UO ---| |---	UO ---| |---	Bits d'état	Bit d'anomalie "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

B Exemples de programmation

B.1 Vue d'ensemble des exemples de programmation

Applications pratiques

Chacune des opérations CONT 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 des opérations CONT :

- 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

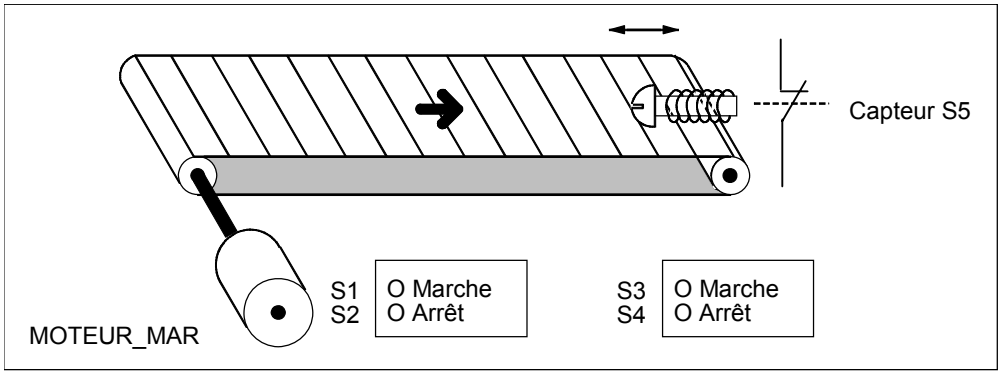
Opérations utilisées

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	Contact à fermeture
—| / |—	Combinaison sur bits	Contact à ouverture
—( )	Combinaison sur bits	Sortie
---(JMPN)	Sauts	Saut à l'intérieur d'un bloc 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

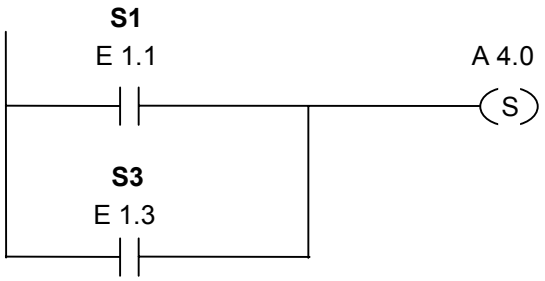
Vous pouvez écrire le programme de commande du tapis roulant en représentant les divers composants du système convoyeur à l'aide **d'adresses absolues** ou à l'aide de **mnémoniques**.

Vous mettez les mnémoniques choisis dans la table des mnémoniques en relation avec les adresses absolues (voir l'aide en ligne de STEP 7).

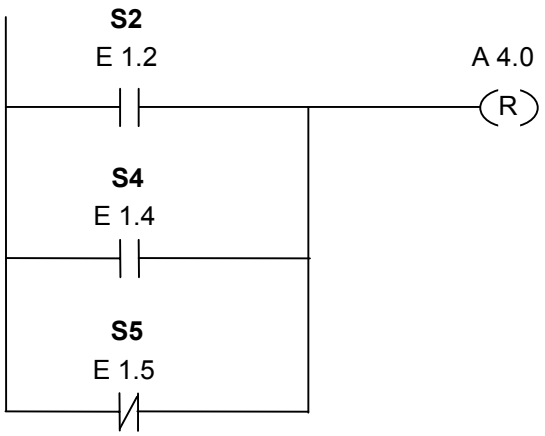
Composant du système	Adresse absolue	Mnémonique	Table de mnémoniques
Bouton-poussoir Marche	E 1.1	S1	E 1.1 S1
Bouton-poussoir Arrêt	E 1.2	S2	E 1.2 S2
Bouton-poussoir Marche	E 1.3	S3	E 1.3 S3
Bouton-poussoir Arrêt	E 1.4	S4	E 1.4 S4
Capteur	E 1.5	S5	E 1.5 S5
Moteur	A 4.0	MOTEUR_MAR	A 4.0 MOTEUR_MAR

Schéma à contacts pour commander un tapis roulant

Réseau 1: Appuyer sur l'un des deux boutons Marche fait démarrer le moteur.

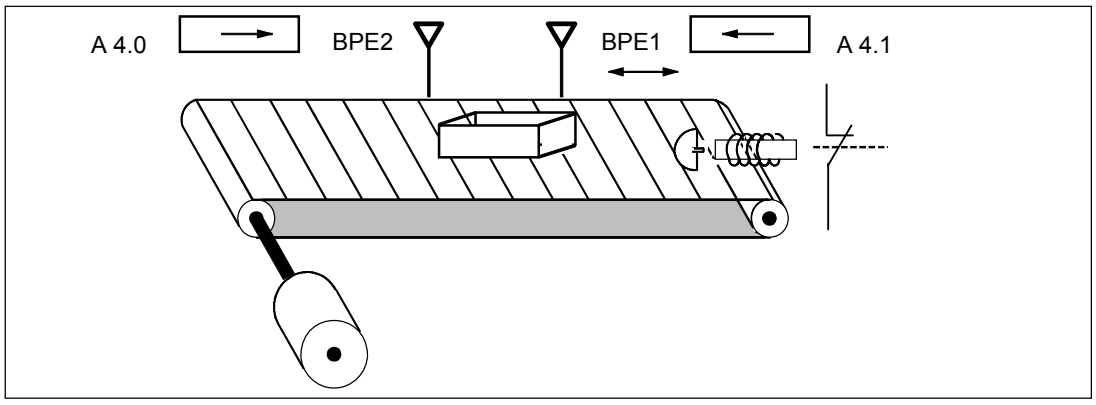


Réseau 2 : Appuyer sur l'un des deux boutons Arrêt ou ouvrir le contact à ouverture à la fin du tapis arrête le moteur.



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

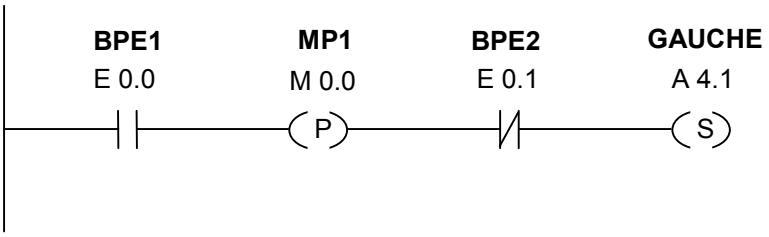
Vous pouvez écrire le programme de commande du tapis roulant en représentant les divers composants du système convoyeur à l'aide **d'adresses absolues** ou à l'aide de **mnémoniques**.

Vous mettez les mnémoniques choisis dans la table des mnémoniques en relation avec les adresses absolues (voir l'aide en ligne de STEP 7).

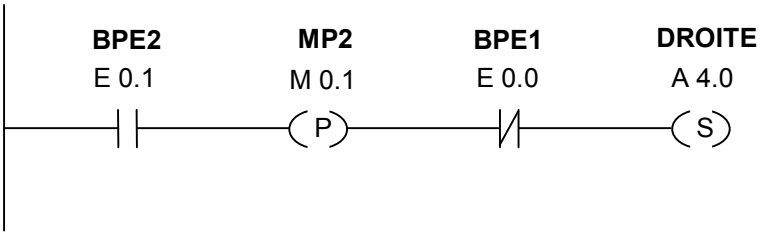
Composant du système	Adresse absolue	Mnémonique	Table de mnémoniques
Barrière photoélectrique 1	E 1.1	BPE1	E 0.0 BPE 1
Barrière photoélectrique 2	E 0.0	BPE2	E 0.1 BPE 2
Affichage pour mouvement vers la droite	A 4.0	DROITE	A 4.0 DROITE
Affichage pour mouvement vers la gauche	A 4.1	GAUCHE	A 4.1 GAUCHE
Mémento de cadence 1	M 0.0	MP1	M 0.0 MP1
Mémento de cadence 2	M 0.1	MP2	M 0.1 MP2

Schéma à contacts pour détecter le sens de déplacement d'un tapis roulant

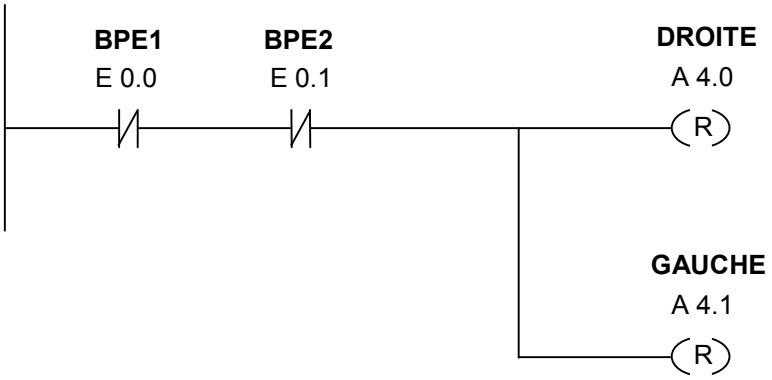
Réseau 1 : Si l'état de signal à l'entrée E 0.0 passe de 0 à 1 (front montant) et si l'état de signal à l'entrée E 0.1 est simultanément à 0, le paquet sur le tapis se déplace vers la gauche.



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 CONT suivant illustre l'utilisation de fonctions de temporisation pour générer une période d'horloge.

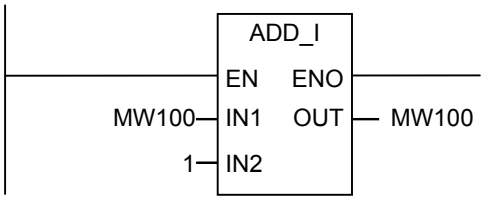
Schéma à contacts pour générer une période d'horloge (rapport d'impulsion 1:1)

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.

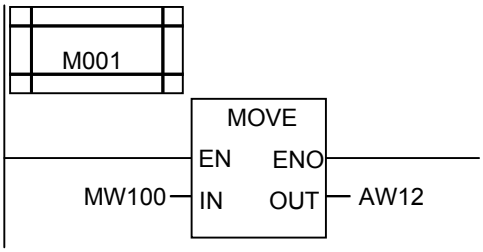
Réseau 2 : L'état de la temporisation est provisoirement mémorisé dans un mémento auxiliaire.

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.

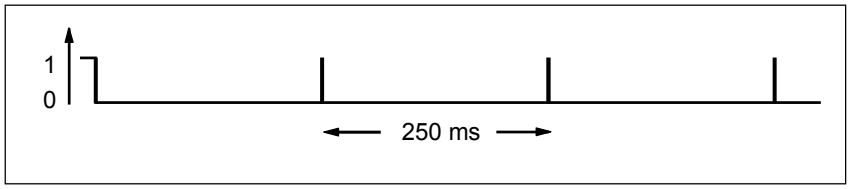


Réseau 5 : L'opération **MOVE** vous permet de voir les différentes fréquences d'horloge aux sorties A 12.0 à A 13.7.



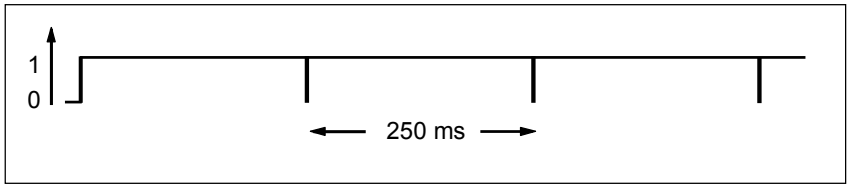
L'interrogation de l'état de signal

L'interrogation du signal de temporisation T1 entraîne le résultat logique suivant pour le contact à ouverture ---I / I--- M0.2 :



La temporisation est redémarrée une fois le temps écoulé. De ce fait, l'interrogation de l'état de signal par l'opération ---I / I--- M0.2 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 :

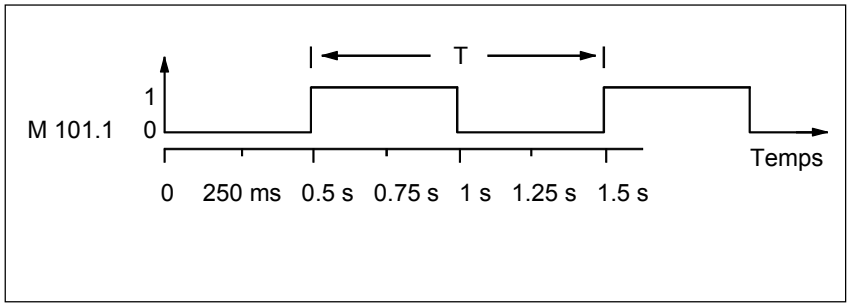
Bits de MB101/ MB100	Fréquence en hertz	Durée	
M 101.0	2.0	0.5 s	(250 ms marche / 250 ms arrêt)
M 101.1	1.0	1 s	(0.5 s marche / 0.5 s arrêt)
M 101.2	0.5	2 s	(1 s marche / 1 s arrêt)
M 101.3	0.25	4 s	(2 s marche / 2 s arrêt)
M 101.4	0.125	8 s	(4 s marche / 4 s arrêt)
M 101.5	0.0625	16 s	(8 s marche / 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.

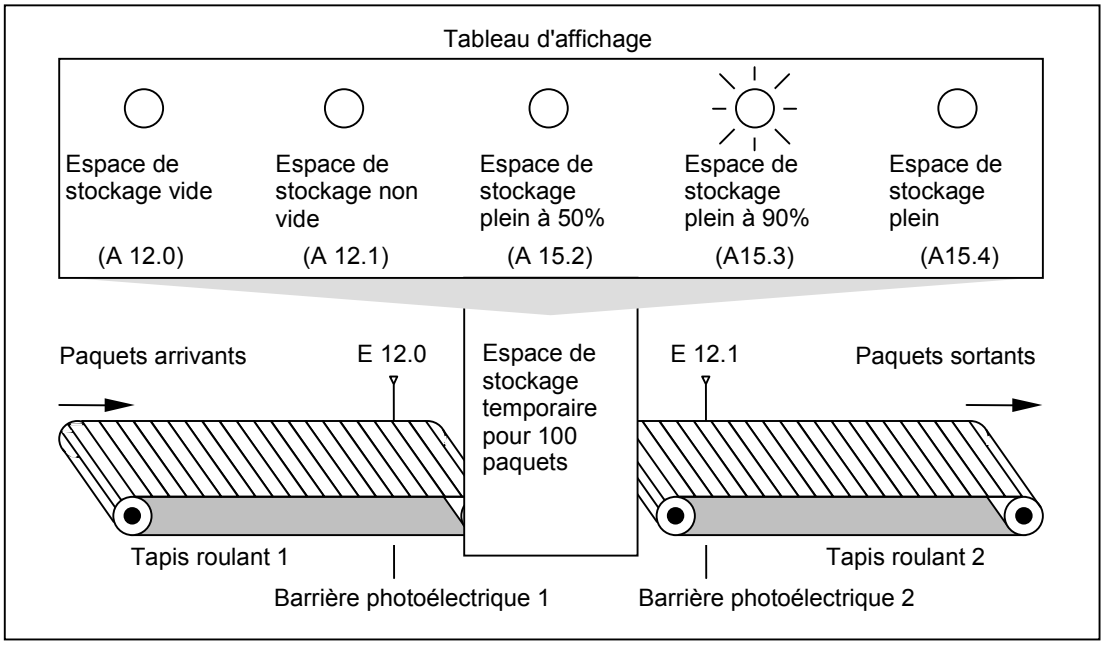
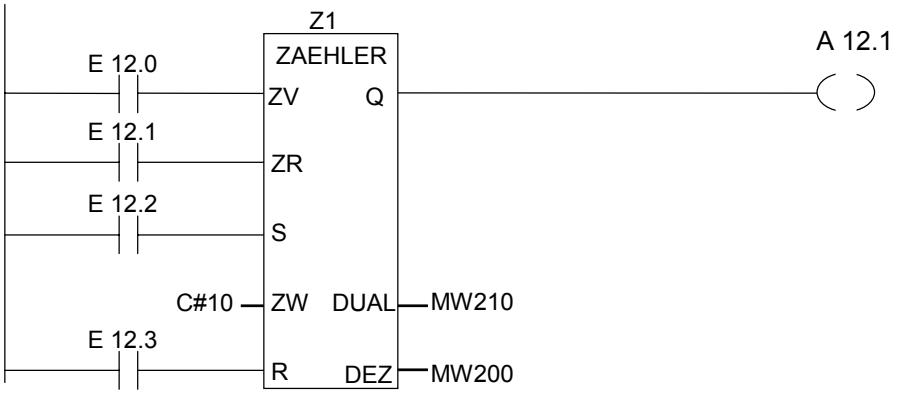


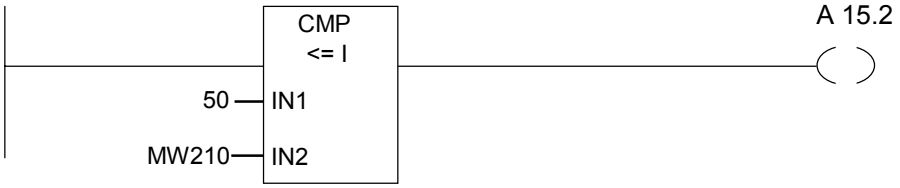
Schéma à contacts pour activer les lampes de signalisation sur un tableau d'affichage

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".

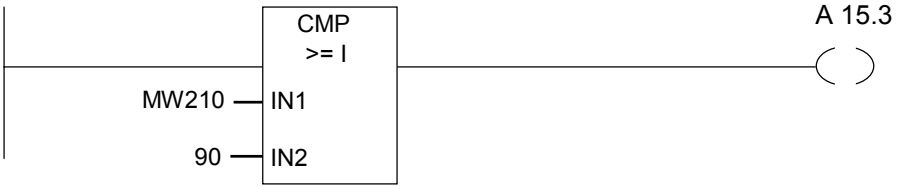


Réseau 2 : La lampe de signalisation A 12.0 indique : "Espace de stockage vide".

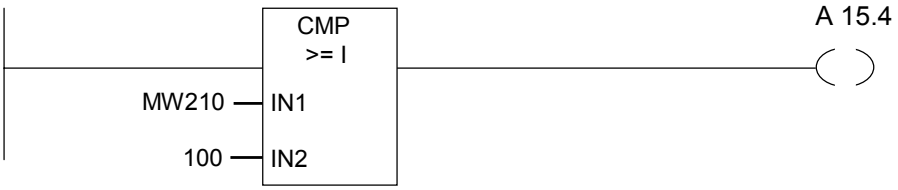
Réseau 3 : Si la valeur 50 est inférieure ou égale à la valeur du compteur (c'est-à-dire que la valeur de comptage est supérieure ou égale à 50), la lampe de signalisation "Espace de stockage plein à 50 %" s'allume.



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

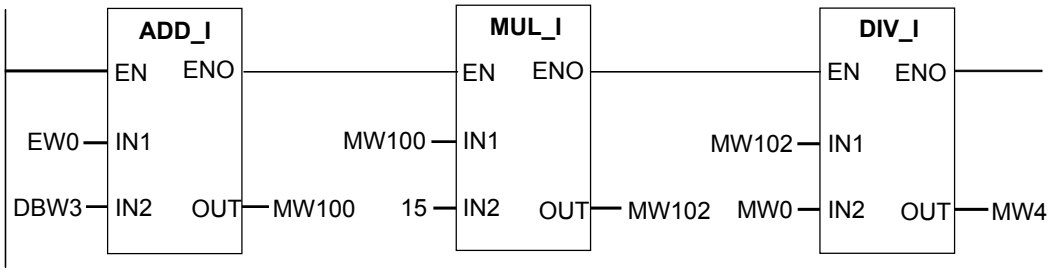
L'exemple de programme suivant montre comment obtenir en utilisant trois opérations arithmétiques sur nombres entiers le même résultat que montre l'équation suivante :

MW4 = ((EW0 + DBW3) x 15) / MW0

Schéma à contacts

Réseau 1 : Ouvrir bloc de données DB1.

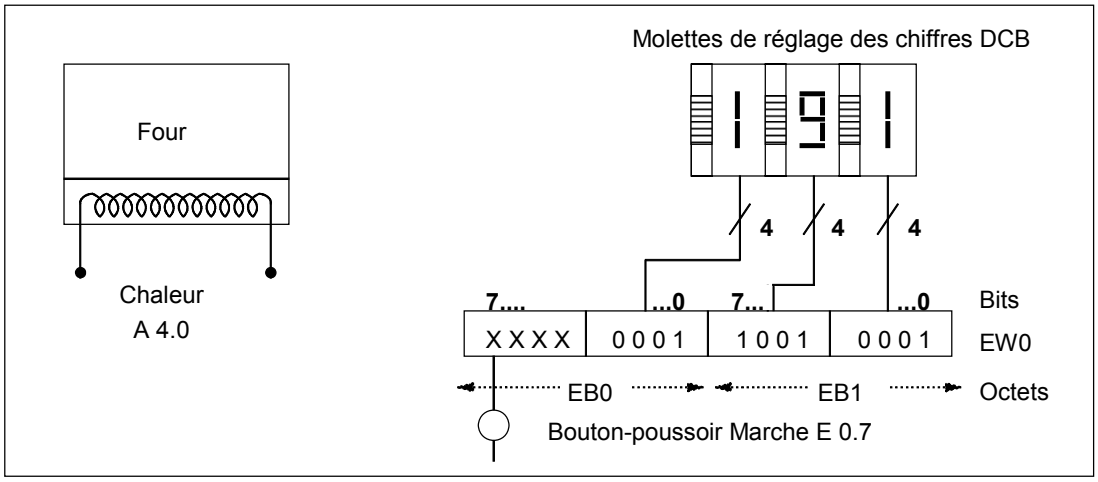
Réseau 2 : Le mot d'entrée EW0 est additionné au mot de données global DBW3 (le bloc de données doit avoir été défini et ouvert) et la somme est chargée dans le mot de mémento MW100. MW100 est ensuite multiplié par 15 et le résultat mémorisé dans le mot de mémento MW102. Puis, MW102 est divisé par MW0 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).



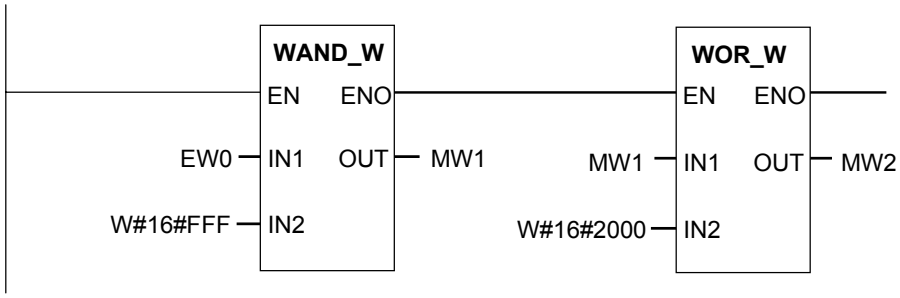
Composants du système	Adresse absolue
Bouton-poussoir Marche	E 0.7
Molette de réglage des unités	E 1.0 à E 1.3
Molette de réglage des dizaines	E 1.4 à E 1.7
Molette de réglage des centaines	E 0.0 à E 0.3
Déclenchement du chauffage	A 4.0

Schéma à contacts

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.


E 0.7
T1
SV
MW2

C Pour travailler en CONT

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

B

C

D

E

F

I

L

M

N

O

P

R

S

T

U

V

W

X

Z