

Modicon M241 Logic Controller

Fonctions et variables système Guide de la bibliothèque PLCSystem

04/2014



Le présent document comprend des descriptions générales et/ou des caractéristiques techniques des produits mentionnés. Il ne peut pas être utilisé pour définir ou déterminer l'adéquation ou la fiabilité de ces produits pour des applications utilisateur spécifiques. Il incombe à chaque utilisateur ou intégrateur de réaliser l'analyse de risques complète et appropriée, l'évaluation et le test des produits pour ce qui est de l'application à utiliser et de l'exécution de cette application. Ni la société Schneider Electric ni aucune de ses sociétés affiliées ou filiales ne peuvent être tenues pour responsables de la mauvaise utilisation des informations contenues dans le présent document. Si vous avez des suggestions, des améliorations ou des corrections à apporter à cette publication, veuillez nous en informer.

Aucune partie de ce document ne peut être reproduite sous quelque forme ou par quelque moyen que ce soit, électronique, mécanique ou photocopie, sans autorisation préalable de Schneider Electric.

Toutes les réglementations de sécurité pertinentes locales doivent être observées lors de l'installation et de l'utilisation de ce produit. Pour des raisons de sécurité et afin de garantir la conformité aux données système documentées, seul le fabricant est habilité à effectuer des réparations sur les composants.

Lorsque des équipements sont utilisés pour des applications présentant des exigences techniques de sécurité, suivez les instructions appropriées.

La non-utilisation du logiciel Schneider Electric ou d'un logiciel approuvé avec nos produits matériels peut entraîner des blessures, des dommages ou un fonctionnement incorrect.

Le non-respect de cette consigne peut entraîner des lésions corporelles ou des dommages matériels.

© 2014 Schneider Electric. Tous droits réservés.

Table des matières



	Consignes de sécurité	7
	A propos de ce manuel	9
Chapitre 1	Variables système du contrôleur M241	13
1.1	Variables système : définition et utilisation	14
	Présentation des variables système	15
	Utilisation des variables système	17
1.2	Structures PLC_R et PLC_W	19
	PLC_R : Variables système en lecture seule du contrôleur	20
	PLC_W : variable système en lecture/écriture de contrôleur	25
1.3	Structures SERIAL_R et SERIAL_W	26
	SERIAL_R[0...1] : variables système en lecture seule de ligne série	27
	SERIAL_W[0...1] : variables système en lecture/écriture de ligne série	28
1.4	Structures ETH_R et ETH_W	29
	ETH_R : variables système en lecture seule du port Ethernet	30
	ETH_W : variables système en lecture/écriture du port Ethernet	35
1.5	Structure TM3_MODULE_R	36
	TM3_MODULE_R[0...13] : Variables système en lecture seule des modules TM3	36
1.6	Structure PROFIBUS_R	37
	PROFIBUS_R : Variables système en lecture seule de PROFIBUS	37
1.7	Structure CART_R	38
	CART_R_STRUCT : Variables système en lecture seule des cartouches	38
Chapitre 2	Fonctions système de M241	39
2.1	Fonctions de lecture de M241	40
	GetImmediateFastInput : Lit l'entrée d'une E/S experte intégrée	41
	GetRtc : Obtenir l'horodatage	42
	IsFirstMastColdCycle : indique si le cycle est le premier cycle MAST après un démarrage à froid	43
	IsFirstMastCycle : indique si le cycle est le premier cycle MAST	44
	IsFirstMastWarmCycle : indique si le cycle est le premier cycle MAST après un démarrage à chaud	46

2.2	Fonctions d'écriture de l'automate M241	47
	PhysicalWriteFastOutputs : Ecrit la sortie rapide d'une E/S experte intégrée	48
	SetRTCDrift : Définir la valeur de compensation de l'horodateur ..	49
2.3	Fonctions utilisateur de M241	51
	DataFileCopy : Commandes de copie de fichier	52
	ExecuteScript : Commandes de script	55
2.4	Fonctions de lecture TM3	57
	TM3_GetModuleBusStatus : Obtient l'état du bus des modules TM3	58
	TM3_GetModuleInternalStatus : Obtient l'état interne des modules TM3	59
Chapitre 3	Types de données de la bibliothèque PLCSystem	
	M241	61
3.1	Types de données des variables système PLC_R/W	62
	PLC_R_APPLICATION_ERROR : Codes d'état des erreurs de l'application détectées	63
	PLC_R_BOOT_PROJECT_STATUS : codes d'état du projet de démarrage	64
	types de données PLC_R_IO_STATUS : codes d'état des E/S	65
	PLC_R_SDCARD_STATUS : Codes d'état de l'emplacement de carte SD	66
	PLC_R_STATUS : codes d'état du contrôleur	67
	PLC_R_STOP_CAUSE : Codes des causes de transition de l'état RUN à un autre	68
	PLC_R_TERMINAL_PORT_STATUS : codes d'état de la connexion du port de programmation	69
	PLC_R_TM3_BUS_STATE : Codes d'état de bus TM3	70
	PLC_W_COMMAND : codes de commande de contrôle	71
3.2	Types de données des variables système DataFileCopy	72
	DataFileCopyError : codes des erreurs détectées	73
	DataFileCopyLocation : Codes d'emplacement	74
3.3	Types de données des variables système ExecScript	75
	ExecuteScriptError : codes des erreurs détectées	75
3.4	Types de données des variables système ETH_R/W	76
	ETH_R_FRAME_PROTOCOL : codes de protocole de transmission de trames	77
	ETH_R_IP_MODE : codes sources d'adresse IP	78
	ETH_R_PORT_DUPLEX_STATUS : codes du mode de transmission	79

ETH_R_PORT_IP_STATUS : codes d'état du port TCP/IP Ethernet .	80
ETH_R_PORT_LINK_STATUS : codes d'état de la liaison de communication	81
ETH_R_PORT_SPEED : codes de la vitesse de communication des ports Ethernet	82
ETH_R_RUN_IDLE : codes d'état fonctionnement et attente Ethernet/IP.	83
3.5 Types de données des variables système TM3_MODULE_R	84
TM3_ERR_CODE : Codes d'erreur des modules d'extension TM3	85
TM3_MODULE_R_ARRAY_TYPE : Type tableau de lecture des modules d'extension TM3	86
TM3_MODULE_STATE : Codes d'état des modules d'extension TM3	87
3.6 Types de données des variables système des cartouches	88
CART_R_ARRAY_TYPE : Type tableau de lecture des cartouches	89
CART_R_MODULE_ID : Identificateur de module de lecture de cartouche	90
CART_R_STATE : Etat de lecture des cartouches	91
3.7 Types de données des fonctions système	92
IMMEDIATE_ERR_TYPE : Codes d'erreur de la fonction <i>GetImmediateFastInput</i> de lecture des entrées d'E/S expertes intégrées	93
RTCSETDRIFT_ERROR : fonction <i>SetRTCDrift</i> - codes des erreurs détectées	94
Annexes	95
Annexe A Représentation des fonctions et blocs fonction	97
Différences entre une fonction et un bloc fonction	98
Utilisation d'une fonction ou d'un bloc fonction en langage IL	99
Utilisation d'une fonction ou d'un bloc fonction en langage ST	103
Glossaire	107
Index	115

Consignes de sécurité



Informations importantes

AVIS

Lisez attentivement ces instructions et examinez le matériel pour vous familiariser avec l'appareil avant de tenter de l'installer, de le faire fonctionner ou d'assurer sa maintenance. Les messages spéciaux suivants que vous trouverez dans cette documentation ou sur l'appareil ont pour but de vous mettre en garde contre des risques potentiels ou d'attirer votre attention sur des informations qui clarifient ou simplifient une procédure.



La présence de ce symbole sur une étiquette "Danger" ou "Avertissement" signale un risque d'électrocution qui provoquera des blessures physiques en cas de non-respect des consignes de sécurité.



Ce symbole est le symbole d'alerte de sécurité. Il vous avertit d'un risque de blessures corporelles. Respectez scrupuleusement les consignes de sécurité associées à ce symbole pour éviter de vous blesser ou de mettre votre vie en danger.

DANGER

DANGER signale un risque qui, en cas de non-respect des consignes de sécurité, **provoque** la mort ou des blessures graves.

AVERTISSEMENT

AVERTISSEMENT signale un risque qui, en cas de non-respect des consignes de sécurité, **peut provoquer** la mort ou des blessures graves.

ATTENTION

ATTENTION signale un risque qui, en cas de non-respect des consignes de sécurité, **peut provoquer** des blessures légères ou moyennement graves.

AVIS

AVIS indique des pratiques n'entraînant pas de risques corporels.

REMARQUE IMPORTANTE

L'installation, l'utilisation, la réparation et la maintenance des équipements électriques doivent être assurées par du personnel qualifié uniquement. Schneider Electric décline toute responsabilité quant aux conséquences de l'utilisation de ce matériel.

Une personne qualifiée est une personne disposant de compétences et de connaissances dans le domaine de la construction, du fonctionnement et de l'installation des équipements électriques, et ayant suivi une formation en sécurité leur permettant d'identifier et d'éviter les risques encourus.

A propos de ce manuel



Présentation

Objectif du document

Ce document est destiné à vous familiariser à l'utilisation des fonctions et variables disponibles dans le Modicon M241 Logic Controller. La bibliothèque PLCSystem du M241 contient des fonctions et des variables permettant de communiquer avec le système contrôleur (réception d'informations et envoi de commandes).

Ce document décrit les types de données, fonctions et variables de la bibliothèque PLCSystem du M241.

Il requiert les connaissances préalables suivantes :

- Connaissances de base sur les fonctionnalités, la structure et la configuration du M241 Logic Controller.
- Programmation en langage FBD, LD, ST, IL ou CFC.
- Variables système (variables globales).

Champ d'application

Le présent document a été mis à jour suite au lancement de SoMachine V4.1.


Document(s) à consulter

Titre de documentation	Référence
SoMachine - Guide de programmation	EIO0000000067 (ENG) ; EIO0000000069 (FRE) ; EIO0000000068 (GER) ; EIO0000000071 (SPA) ; EIO0000000070 (ITA) ; EIO0000000072 (CHS)
Modicon M241 Logic Controller - Guide de référence du matériel	EIO0000001456 (ENG); EIO0000001457 (FRE); EIO0000001458 (GER); EIO0000001459 (SPA); EIO0000001460 (ITA); EIO0000001461 (CHS)

Titre de documentation	Référence
Modicon M241 Logic Controller - Guide de programmation	EIO0000001432 (ENG) ; EIO0000001433 (FRE) ; EIO0000001434 (GER) ; EIO0000001435 (SPA) ; EIO0000001436 (ITA) ; EIO0000001437 (CHS)

Vous pouvez télécharger ces publications et autres informations techniques depuis notre site web à l'adresse : www.schneider-electric.com.

Information spécifique au produit

 AVERTISSEMENT
<p>PERTE DE CONTROLE</p> <ul style="list-style-type: none"> ● Le concepteur d'un circuit de commande doit tenir compte des modes de défaillance potentiels des canaux de commande et, pour certaines fonctions de commande critiques, prévoir un moyen d'assurer la sécurité en maintenant un état sûr pendant et après la défaillance. Par exemple, l'arrêt d'urgence, l'arrêt en cas de surcourse, la coupure de courant et le redémarrage sont des fonctions de commande cruciales. ● Des canaux de commande séparés ou redondants doivent être prévus pour les fonctions de commande critiques. ● Les liaisons de communication peuvent faire partie des canaux de commande du système. Une attention particulière doit être prêtée aux implications des délais de transmission non prévus ou des pannes de la liaison. ● Respectez toutes les réglementations de prévention des accidents ainsi que les consignes de sécurité locales.¹ ● Chaque implémentation de cet équipement doit être testée individuellement et entièrement pour s'assurer du fonctionnement correct avant la mise en service. <p>Le non-respect de ces instructions peut provoquer la mort, des blessures graves ou des dommages matériels.</p>

¹ Pour plus d'informations, consultez le document NEMA ICS 1.1 (dernière édition), « Safety Guidelines for the Application, Installation, and Maintenance of Solid State Control » (Directives de sécurité pour l'application, l'installation et la maintenance de commande statique) et le document NEMA ICS 7.1 (dernière édition), « Safety Standards for Construction and Guide for Selection, Installation, and Operation of Adjustable-Speed Drive Systems » (Normes de sécurité relatives à la construction et manuel de sélection, installation et opération de variateurs de vitesse) ou son équivalent en vigueur dans votre pays.

AVERTISSEMENT

COMPORTEMENT IMPREVU DE L'EQUIPEMENT

- N'utilisez que le logiciel approuvé par Schneider Electric pour faire fonctionner cet équipement.
- Mettez à jour votre programme d'application chaque fois que vous modifiez la configuration matérielle physique.

Le non-respect de ces instructions peut provoquer la mort, des blessures graves ou des dommages matériels.

Chapitre 1

Variables système du contrôleur M241

Présentation

Ce chapitre :

- fournit une introduction aux variables système (*voir page 14*) ;
- décrit les variables système (*voir page 20*) disponibles avec la bibliothèque PLCSystem de M241.

Contenu de ce chapitre

Ce chapitre contient les sous-chapitres suivants :

Sous-chapitre	Sujet	Page
1.1	Variables système : définition et utilisation	14
1.2	Structures PLC_R et PLC_W	19
1.3	Structures SERIAL_R et SERIAL_W	26
1.4	Structures ETH_R et ETH_W	29
1.5	Structure TM3_MODULE_R	36
1.6	Structure PROFIBUS_R	37
1.7	Structure CART_R	38

Sous-chapitre 1.1

Variables système : définition et utilisation

Présentation

Cette section définit les variables système et explique leur mise en œuvre dans le Modicon M241 Logic Controller.

Contenu de ce sous-chapitre

Ce sous-chapitre contient les sujets suivants :

Sujet	Page
Présentation des variables système	15
Utilisation des variables système	17

Présentation des variables système

Introduction

Cette section décrit comment les variables système sont mises en œuvre. Les variables système :

- permettent d'accéder à des informations générales sur le système, de réaliser des diagnostics système et de commander des actions simples ;
- sont des variables structurées selon les définitions et conventions de désignation de la norme CEI 61131-3. Vous pouvez accéder aux variables système à l'aide du nom symbolique CEI `PLC_GVL`. Certaines variables `PLC_GVL` sont en lecture seule (par exemple, `PLC_R`) et d'autres sont en lecture-écriture (par exemple, `PLC_W`).
- sont déclarées automatiquement comme des variables globales. Elles s'appliquent à l'ensemble du système et toute POU (unité organisationnelle de programme) d'une tâche peut y accéder.

Convention de désignation

Les variables système sont identifiées par :

- un nom de structure qui représente la catégorie de variables système. Par exemple, `PLC_R` représente un nom de structure de variables en lecture seule utilisées pour le diagnostic du contrôleur.
- un ensemble de noms de composant qui identifie le rôle de la variable. Par exemple, `i_wVendorID` représente l'ID du fournisseur du contrôleur.

Vous pouvez accéder aux variables système en entrant leur nom de structure suivi du nom du composant.

Voici un exemple de mise en œuvre de variables système :

```
VAR
    myCtr_Serial : DWORD;
    myCtr_ID : DWORD;
    myCtr_FramesRx : UDINT;
END_VAR

myCtr_Serial := PLC_R.i_dwSerialNumber;
myCtr_ID := PLC_R.i_wVendorID;
myCtr_FramesRx := SERIAL_R[0].i_udiFramesReceivedOK;
```

NOTE : Dans l'exemple ci-dessus, le nom complet de la variable système est `PLC_GVL.PLC_R.i_wVendorID`. Le `PLC_GVL` est implicite lors de la déclaration d'une variable à l'aide de l'**Aide à la saisie**, mais vous pouvez aussi l'entrer en intégralité. Les bonnes pratiques de programmation préconisent souvent d'utiliser le nom complet de la variable dans les déclarations.

Emplacement des variables système

Deux sortes de variables système sont définies pour la programmation du contrôleur :

- variables localisées
- variables non localisées

Les variables localisées :

- ont un emplacement fixe dans une zone %MW statique : %MW60000 à %MW60199 pour les variables système en lecture seule ;
- sont accessibles par l'intermédiaire de requêtes Modbus TCP, Modbus série et EtherNet/IP dans les états RUNNING et STOPPED ;
- sont utilisées dans des programmes SoMachine conformément à la convention `structure_name.component_name` expliquée précédemment. Les adresses %MW de 0 to 59999 sont accessibles directement. Les adresses supérieures sont considérées hors plage par SoMachine et sont uniquement accessibles via la convention `structure_name.component_name`.

Les variables non affectées :

- ne se trouvent pas physiquement dans la zone %MW ;
- ne sont pas accessibles par le biais de requêtes de bus de terrain ou de réseau, sauf si vous les cherchez dans la table de réaffectation, et elles ne sont alors accessibles que dans les états RUNNING et STOPPED ; La table de réaffectation utilise les zones %MW dynamiques suivantes :
 - %MW60200 à %MW61999 pour les variables système en lecture seule,
 - %MW62200 à %MW63999 pour les variables en lecture/écriture.
- sont utilisées dans des programmes SoMachine conformément à la convention `structure_name.component_name` expliquée précédemment. Les adresses %MW de 0 à 59999 sont accessibles directement. Les adresses supérieures sont considérées hors plage par SoMachine et sont uniquement accessibles via la convention `structure_name.component_name`.

Utilisation des variables système

Introduction

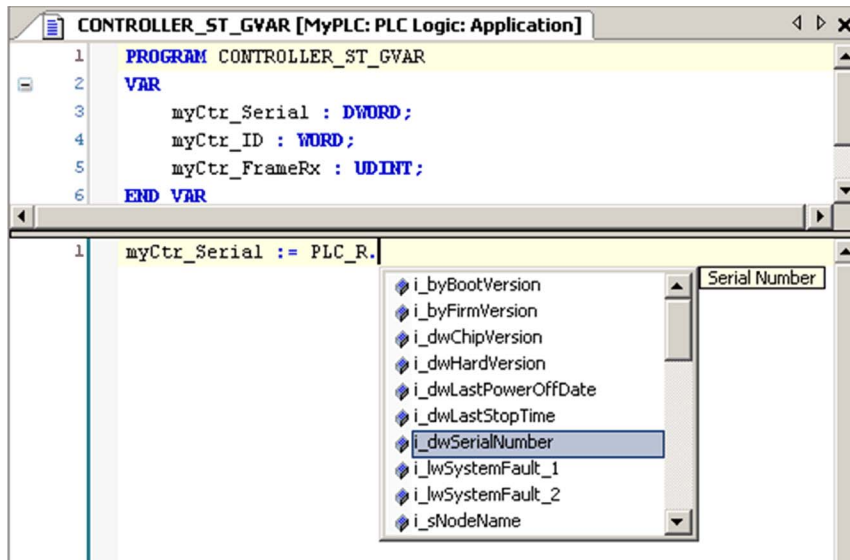
Cette rubrique décrit la procédure de programmation et d'utilisation des variables système dans SoMachine.

Les variables système ont un champ d'application global et vous pouvez les utiliser dans tous les POU (unité organisationnelle de programme) de l'application.

Il n'est pas nécessaire de déclarer les variables système dans la liste des variables globales (GVL). Elles sont déclarées automatiquement à partir de la bibliothèque système du contrôleur.

Utilisation des variables système dans un POU

SoMachine a une fonction de saisie automatique. Dans un **POU**, commencez par entrer le nom de structure de la variable système (PLC_R, PLC_W, ...) suivi d'un point. Les variables système s'affichent dans l'**Aide à la saisie**. Vous pouvez sélectionner la variable de votre choix ou entrer manuellement son nom en intégralité.



NOTE : Dans l'exemple ci-dessus, une fois que le nom de structure `PLC_R.` a été entré, SoMachine affiche un menu contextuel des noms de composants/variables possibles.

Exemple

L'exemple ci-dessous décrit l'utilisation de certaines variables système :

```
VAR myCtr_Serial : DWORD; myCtr_ID : WORD; myCtr_FramesRx : UDINT;  
END_VAR  
  
myCtr_Serial := PLC_R.i_dwSerialNumber; myCtr_ID := PLC_R.i_wVendorID;  
myCtr_FramesRx := SERIAL_R[0].i_udiFramesReceivedOK;
```

Sous-chapitre 1.2

Structures PLC_R et PLC_W

Présentation

Cette section répertorie et décrit les variables système incluses dans les structures PLC_R et PLC_W.

Contenu de ce sous-chapitre

Ce sous-chapitre contient les sujets suivants :

Sujet	Page
PLC_R : Variables système en lecture seule du contrôleur	20
PLC_W : variable système en lecture/écriture de contrôleur	25

PLC_R : Variables système en lecture seule du contrôleur

Structure de la variable

Le tableau suivant décrit les paramètres de la variable système PLC_R (type PLC_R_STRUCT) :

Adresse Modbus ⁽¹⁾	Nom de la variable	Type	Commentaire
60000	i_wVendorID	WORD	ID du fournisseur du contrôleur. 101A hex = Schneider Electric
60001	i_wProductID	WORD	ID de référence du contrôleur. NOTE : L'ID du fournisseur et l'ID de référence constituent l'ID cible du contrôleur, indiqué dans l'écran des paramètres de communication (ID cible = 101A XXXX hex).
60002	i_dwSerialNumber	DWORD	Numéro de série du contrôleur.
60004	i_byFirmVersion	ARRAY[0..3] OF BYTE	Version du micrologiciel du contrôleur [aa.bb.cc.dd] : <ul style="list-style-type: none"> ● i_byFirmVersion[0] = aa ● ... ● i_byFirmVersion[3] = dd
60006	i_byBootVersion	ARRAY[0..3] OF BYTE	Version du démarrage du contrôleur [aa.bb.cc.dd] : <ul style="list-style-type: none"> ● i_byBootVersion[0] = aa ● ... ● i_byBootVersion[3] = dd
60008	i_dwHardVersion	DWORD	Version du matériel du contrôleur.
60010	i_dwChipVersion	DWORD	Version du coprocesseur du contrôleur.
60012	i_wStatus	PLC_R_STATUS (voir page 67)	Etat du contrôleur.
60013	i_wBootProjectStatus	PLC_R_BOOT_PROJECT_STATUS (voir page 64)	Renvoi des informations sur l'application de démarrage stockée dans la mémoire Flash.
60014	i_wLastStopCause	PLC_R_STOP_CAUSE (voir page 68)	Cause du dernier passage du mode d'exécution (RUN) à un autre état.
60015	i_wLastApplicationError	PLC_R_APPLICATION_ERROR (voir page 63)	Cause de la dernière exception du contrôleur.

Adresse Modbus ⁽¹⁾	Nom de la variable	Type	Commentaire
60016	i_lwSystemFault_1	LWORD	<p>Le champ de bits FFFF FFFF FFFF FFFF hex indique qu'aucune erreur n'a été détectée.</p> <p>Un bit de niveau bas signifie qu'une erreur a été détectée :</p> <ul style="list-style-type: none"> ● bit 0 = erreur d'E/S experte détectée ● bit 1 = erreur TM3 détectée ● bit 2 = erreur IF1 Ethernet détectée ● bit 3 = erreur IF2 Ethernet détectée ● bit 4 = erreur de surintensité détectée sur ligne série 1 ● bit 5 = erreur détectée sur ligne série 2 ● bit 6 = erreur CAN 1 détectée ● bit 7 = erreur de cartouche 1 détectée ● bit 8 = erreur de cartouche 2 détectée ● bit 9 = erreur TM4 détectée ● bit 10 = erreur de carte SD détectée ● bit 11 = erreur de pare-feu détectée
60020	i_lwSystemFault_2	LWORD	<p>Le champ de bits FFFF hex indique qu'aucune erreur n'a été détectée.</p> <p>Si i_wIOStatus1 = PLC_R_IO_SHORTCUT_FAULT, i_lwSystemFault_2 signifie :</p> <ul style="list-style-type: none"> ● bit 0 = 0 : court-circuit détecté dans le bloc PTO0 ● bit 1 = 0 : court-circuit détecté dans le bloc PTO1 ● bit 2 = 0 : court-circuit détecté dans le groupe de sorties 1 ● bit 3 = 0 : court-circuit détecté dans le groupe de sorties 2 ● bit 4 = 0 : court-circuit détecté dans le groupe de sorties 3
60024	i_wIOStatus1	PLC_R_IO_STATUS (voir page 65)	Etat des E/S expertes intégrées.
60025	i_wIOStatus2	PLC_R_IO_STATUS (voir page 65)	Etat d'E/S TM3.

Adresse Modbus ⁽¹⁾	Nom de la variable	Type	Commentaire
60026	i_wClockBatteryStatus	WORD	Etat de la batterie de l'horodateur : <ul style="list-style-type: none"> ● 0 = changement de batterie requis ● 100 = batterie en pleine charge Les autres valeurs (1 à 99) représentent le pourcentage de charge. Par exemple, si la valeur est 75, la batterie est chargée à 75 %.
60028	i_dwAppliSignature1	DWORD	Premier des 4 DWORD de la signature (16 octets au total). La signature de l'application est générée par le logiciel pendant la construction.
60030	i_dwAppliSignature2	DWORD	Deuxième des 4 DWORD de la signature (16 octets au total). La signature de l'application est générée par le logiciel pendant la construction.
60032	i_dwAppliSignature3	DWORD	Troisième des 4 DWORD de la signature (16 octets au total). La signature de l'application est générée par le logiciel pendant la construction.
60034	i_dwAppliSignature4	DWORD	Quatrième des 4 DWORD de la signature (16 octets au total). La signature de l'application est générée par le logiciel pendant la construction.
⁽¹⁾ Inaccessible via l'application.			

s/o	i_sVendorName	STRING (31)	Nom du fournisseur : « Schneider Electric ».
s/o	i_sProductRef	STRING (31)	Référence du contrôleur.
s/o	i_sNodeName	STRING (99)	Nom du nœud sur le réseau SoMachine.
s/o	i_dwLastStopTime	DWORD	Heure du dernier arrêt détecté, en secondes depuis le 1er janvier 1970 à 00:00:00 (UTC).

s/o	i_dwLastPowerOffDate	DWORD	Date et heure de la dernière mise hors tension détectée, en secondes depuis le 1er janvier 1970 à 00:00:00 (UTC). NOTE : Convertissez cette valeur en date et heure avec la fonction <code>SysTimeRtcConvertUtcToDate</code> . Pour plus d'informations sur la conversion Date et heure, reportez-vous au Guide de la bibliothèque <code>SysTime</code> (voir <i>SoMachine, Affichage et réglage de l'horodateur, Guide de la bibliothèque SysTime</i>).
s/o	i_uiEventsCounter	UINT	Nombre d'événements externes détectés sur des entrées configurées pour la détection d'événements externes depuis le dernier démarrage à froid. Effectuez la réinitialisation par un démarrage à froid ou en exécutant la commande <code>PLC_W.q_wResetCounterEvent</code> .
s/o	i_wTerminalPortStatus	PLC_R_TERMINAL_PORT_STATUS (voir page 69)	Etat du port de programmation USB (USB mini B).
s/o	i_wSdCardStatus	PLC_R_SDCARD_STATUS (voir page 66)	Etat de la carte SD.
s/o	i_wUsrFreeFileHdl	WORD	Nombre de descripteurs de fichier disponibles. Un descripteur de fichier correspond à la ressource allouée par le système lorsque vous ouvrez un fichier.
s/o	i_udiUsrFsTotalBytes	UDINT	Taille de la mémoire totale du système de fichiers de l'utilisateur (en octets). Correspond à la taille de la mémoire Flash du répertoire « /usr/ ».
s/o	i_udiUsrFsFreeBytes	UDINT	Taille de la mémoire libre du système de fichiers de l'utilisateur (en octets).

s/o	i_uiTM3BusState	PLC_R_TM3_BUS_STATE (voir page 70)	Etat du bus TM3. i_uiTM3BusState peut avoir les valeurs suivantes : <ul style="list-style-type: none"> ● 1 : TM3_CONF_ERROR La configuration physique et la configuration SoMachine ne correspondent pas. ● 3 : TM3_OK La configuration physique correspond à la configuration SoMachine. ● 4 : TM3_POWER_SUPPLY_ERROR Le bus TM3 n'est pas alimenté (par exemple, lorsque le contrôleur est alimenté par USB).
s/o	i_ExpertIO_RunStop_Input	BYTE	L'emplacement de l'entrée Run/Stop est : <ul style="list-style-type: none"> ● 16..FF hex si l'E/S experte n'est pas configurée ● 0 pour %IX0.0 ● 1 pour %IX0.1
s/o	i_x10msClk	BOOL	Bit de base de temps : 10 ms. Cette variable s'active et se désactive par période de 10 ms. La valeur bascule lorsque le contrôleur logique est dans l'état Stop et dans l'état Run.
s/o	i_x100msClk	BOOL	Bit de base de temps : 100 ms. Cette variable s'active et se désactive par période de 100 ms. La valeur bascule lorsque le contrôleur logique est dans l'état Stop et dans l'état Run.
s/o	i_x1sClk	BOOL	Bit de base de temps : 1 s. Cette variable s'active et se désactive par période de 1 s. La valeur bascule lorsque le contrôleur logique est dans l'état Stop et dans l'état Run.

NOTE : s/o signifie qu'aucun mappage d'adresse Modbus n'est prédéfini pour cette variable système.

PLC_W : variable système en lecture/écriture de contrôleur

Structure de la variable

Ce tableau décrit les paramètres de la variable système PLC_W (type PLC_W_STRUCT) :

%MW	Nom de la variable	Type	Commentaire
s/o	q_wResetCounterEvent	WORD	Le passage de 0 à 1 réinitialise le compteur d'événements (PLC_R.i_uiEventsCounter). Pour réinitialiser à nouveau le compteur, il est nécessaire de mettre ce registre à 0 de sorte qu'un nouveau passage de 0 à 1 puisse intervenir.
s/o	q_uiOpenPLCControl	UINT	Lorsque la valeur passe de 0 à 6699, la commande inscrite précédemment dans la variable PLC_W.q_wPLCControl suivante est exécutée.
s/o	q_wPLCControl	PLC_W_COMMAND (voir page 71)	La commande RUN/STOP du contrôleur est exécutée lorsque la valeur de la variable système PLC_R.q_uiOpenPLCControl passe de 0 à 6699.

NOTE : s/o signifie qu'aucun mappage %MW n'est prédéfini pour cette variable système.

Sous-chapitre 1.3

Structures SERIAL_R et SERIAL_W

Présentation

Cette section répertorie et décrit les variables système des structures SERIAL_R et SERIAL_W.

Contenu de ce sous-chapitre

Ce sous-chapitre contient les sujets suivants :

Sujet	Page
SERIAL_R[0...1] : variables système en lecture seule de ligne série	27
SERIAL_W[0...1] : variables système en lecture/écriture de ligne série	28

SERIAL_R[0...1] : variables système en lecture seule de ligne série

Introduction

SERIAL_R est un tableau de 2 éléments de type SERIAL_R_STRUCT. Chaque élément du tableau renvoie des variables système de diagnostic pour la ligne série correspondante.

Pour le M241 Logic Controller :

- Serial_R[0] désigne la ligne série 1.
- Serial_R[1] désigne la ligne série 2.

Structure de la variable

Ce tableau décrit les paramètres des variables système SERIAL_R[0...1] :

%MW	Nom de la variable	Type	Commentaire
Ligne série			
s/o	i_udiFramesTransmittedOK	UDINT	Nombre de trames transmises avec succès.
s/o	i_udiFramesReceivedOK	UDINT	Nombre de trames reçues sans aucune erreur détectée.
s/o	i_udiRX_MessagesError	UDINT	Nombre de trames reçues avec erreurs détectées (somme de contrôle, parité).
Spécifique Modbus			
s/o	i_uiSlaveExceptionCount	UINT	Nombre de réponses d'exception Modbus renvoyées par le Logic Controller.
s/o	i_udiSlaveMsgCount	UINT	Nombre de messages reçus du maître et adressés au Logic Controller.
s/o	i_uiSlaveNoRespCount	UINT	Nombre de demandes de diffusion Modbus reçues par le Logic Controller.
s/o	i_uiSlaveNakCount	UINT	Non utilisée
s/o	i_uiSlaveBusyCount	UINT	Non utilisée
s/o	i_uiCharOverrunCount	UINT	Nombre de débordements de caractères.
s/o signifie qu'aucun mappage %MW n'est prédéfini pour cette variable système. Non utilisée signifie que la variable n'est pas gérée par le système et que si sa valeur est différente de zéro, elle doit être considérée comme variable parasite.			

Les compteurs SERIAL_R sont réinitialisés en cas de :

- téléchargement,
- réinitialisation du contrôleur ;
- commande SERIAL_W[x].q_wResetCounter ;
- commande de réinitialisation associée au code fonction n° 8 de la requête Modbus.

SERIAL_W[0...1] : variables système en lecture/écriture de ligne série

Introduction

SERIAL_W est un tableau de 2 éléments de type SERIAL_W_STRUCT. Chaque élément du tableau réinitialise les variables système SERIAL_R de la ligne série correspondante.

Pour le M241 Logic Controller :

- Serial_W[0] désigne la ligne série 1.
- Serial_W[1] désigne la ligne série 2.

Structure de la variable

Ce tableau décrit les paramètres de la variable système SERIAL_W[0...1] :

%MW	Nom de la variable	Type	Commentaire
s/o	q_wResetCounter	WORD	Le passage de 0 à 1 réinitialise tous les compteurs SERIAL_R[0...1]. Pour réinitialiser à nouveau les compteurs, il est nécessaire de mettre ce registre à 0 de sorte qu'un nouveau passage de 0 à 1 puisse intervenir.

NOTE : s/o signifie qu'aucun mappage %MW n'est prédéfini pour cette variable système.

Sous-chapitre 1.4

Structures ETH_R et ETH_W

Présentation

Cette section répertorie et décrit les variables système incluses dans les structures ETH_R et ETH_W.

Contenu de ce sous-chapitre

Ce sous-chapitre contient les sujets suivants :

Sujet	Page
ETH_R : variables système en lecture seule du port Ethernet	30
ETH_W : variables système en lecture/écriture du port Ethernet	35

ETH_R : variables système en lecture seule du port Ethernet

Structure de la variable

Le tableau suivant décrit les paramètres de la variable système ETH_R (type ETH_R_STRUCT) :

%MW	Nom de la variable	Type	Commentaire
60050	i_byIPAddress	ARRAY[0..3] OF BYTE	Adresse IP [aaa.bbb.ccc.ddd] : <ul style="list-style-type: none"> ● i_byIPAddress[0] = aaa ● ... ● i_byIPAddress[3] = ddd
60052	i_bySubNetMask	ARRAY[0..3] OF BYTE	Masque de sous-réseau [aaa.bbb.ccc.ddd] : <ul style="list-style-type: none"> ● i_bySub-netMask[0] = aaa ● ... ● i_bySub-netMask[3] = ddd
60054	i_byGateway	ARRAY[0..3] OF BYTE	Adresse de la passerelle [aaa.bbb.ccc.ddd] : <ul style="list-style-type: none"> ● i_byGateway[0] = aaa ● ... ● i_byGateway[3] = ddd
60056	i_byMACAddress	ARRAY[0..5] OF BYTE	Adresse MAC [aa.bb.cc.dd.ee.ff] : <ul style="list-style-type: none"> ● i_byMACAddress[0] = aa ● ... ● i_byMACAddress[5] = ff
60059	i_sDeviceName	STRING(15)	Nom utilisé pour obtenir l'adresse IP auprès du serveur.
s/o	i_wIpMode	ETH_R_IP_MODE (voir page 78)	Méthode utilisée pour obtenir une adresse IP.
s/o	i_byFDRServerIPAddress	ARRAY[0..3] OF BYTE	Adresse IP [aaa.bbb.ccc.ddd] du serveur DHCP ou BootP : <ul style="list-style-type: none"> ● i_byFDRServerIPAddress[0] = aaa ● ... ● i_byFDRServerIPAddress[3] = ddd Egale à 0.0.0.0 en cas d'utilisation d'une adresse IP enregistrée ou par défaut.
s/o	i_udiOpenTcpConnections	UDINT	Nombre de connexions TCP ouvertes.
s/o signifie qu'aucun mappage %MW n'est prédéfini pour cette variable système.			

%MW	Nom de la variable	Type	Commentaire
s/o	i_udiFramesTransmittedOK	UDINT	Nombre de trames transmises correctement. Réinitialisation lors de la mise sous tension ou avec la commande de réinitialisation ETH_W.q_wResetCounter.
s/o	i_udiFramedReceivedOK	UDINT	Nombre de trames reçues correctement. Réinitialisation lors de la mise sous tension ou avec la commande de réinitialisation ETH_W.q_wResetCounter.
s/o	i_udiTransmitBufferErrors	UDINT	Nombre de trames transmises avec détection d'erreurs. Réinitialisation lors de la mise sous tension ou avec la commande de réinitialisation ETH_W.q_wResetCounter.
s/o	i_udiReceiveBufferErrors	UDINT	Nombre de trames reçues avec détection d'erreurs. Réinitialisation lors de la mise sous tension ou avec la commande de réinitialisation ETH_W.q_wResetCounter.
s/o	i_wFrameSendingProtocol	ETH_R_FRAME_PROTOCOL (voir page 77)	Protocole Ethernet configuré pour l'envoi des trames (IEEE 802.3 ou Ethernet II).
s/o	i_wPortALinkStatus	ETH_R_PORT_LINK_STATUS (voir page 81)	Liaison du port Ethernet (0 = aucune liaison, 1 = liaison connectée à un autre équipement Ethernet).
s/o	i_wPortASpeed	ETH_R_PORT_SPEED (voir page 82)	Débit réseau du port Ethernet (10 ou 100 Mbits/s).
s/o	i_wPortADuplexStatus	ETH_R_PORT_DUPLEX_STATUS (voir page 79)	Etat duplex du port Ethernet (0 = semi duplex ou 1 = duplex intégral).
s/o	i_udiPortACollisions	UDINT	Nombre de trames impliquées dans une ou plusieurs collisions et transmises correctement par la suite. Réinitialisation lors de la mise sous tension ou avec la commande de réinitialisation ETH_W.q_wResetCounter.
Spécifique à Modbus TCP/IP			
s/o signifie qu'aucun mappage %MW n'est prédéfini pour cette variable système.			

%MW	Nom de la variable	Type	Commentaire
s/o	i_udiModbusMessageTransmitted	UDINT	Nombre de messages Modbus transmis. Réinitialisation lors de la mise sous tension ou avec la commande de réinitialisation ETH_W.q_wResetCounter.
s/o	i_udiModbusMessageReceived	UDINT	Nombre de messages Modbus reçus. Réinitialisation lors de la mise sous tension ou avec la commande de réinitialisation ETH_W.q_wResetCounter.
s/o	i_udiModbusErrorMessage	UDINT	Messages de détection d'erreurs Modbus transmis et reçus. Réinitialisation lors de la mise sous tension ou avec la commande de réinitialisation ETH_W.q_wResetCounter.
s/o signifie qu'aucun mappage %MW n'est prédéfini pour cette variable système.			

%MW	Nom de la variable	Type	Commentaire
Spécifique à EtherNet/IP			
s/o	i_udiETHIP_IOMessagingTransmitted	UDINT	Trames EtherNet/IP de classe 1 transmises. Réinitialisation lors de la mise sous tension ou avec la commande de réinitialisation ETH_W.q_wResetCounter.
s/o	i_udiETHIP_IOMessagingReceived	UDINT	Trames EtherNet/IP de classe 1 reçues. Réinitialisation lors de la mise sous tension ou avec la commande de réinitialisation ETH_W.q_wResetCounter.
s/o	i_udiUCMM_Request	UDINT	Messages EtherNet/IP non connectés reçus. Réinitialisation lors de la mise sous tension ou avec la commande de réinitialisation ETH_W.q_wResetCounter.
s/o signifie qu'aucun mappage %MW n'est prédéfini pour cette variable système. Non utilisée signifie que la variable n'est pas gérée par le système et que si sa valeur est différente de zéro, elle doit être considérée comme variable parasite.			

%MW	Nom de la variable	Type	Commentaire
s/o	i_udiUCMM_Error	UDINT	Messages EtherNet/IP non connectés non valides reçus. Réinitialisation lors de la mise sous tension ou avec la commande de réinitialisation ETH_W.q_wResetCounter.
s/o	i_udiClass3_Request	UDINT	Requêtes EtherNet/IP de classe 3 reçues. Réinitialisation lors de la mise sous tension ou avec la commande de réinitialisation ETH_W.q_wResetCounter.
s/o	i_udiClass3_Error	UDINT	Requêtes EtherNet/IP de classe 3 non valides reçues. Réinitialisation lors de la mise sous tension ou avec la commande de réinitialisation ETH_W.q_wResetCounter.
s/o	i_uiAssemblyInstanceInput	UINT	Numéro de l'instance d'assemblage d'entrée. Pour plus d'informations, consultez le guide de programmation de votre contrôleur.
s/o	i_uiAssemblyInstanceInputSize	UINT	Taille de l'instance d'assemblage d'entrée. Pour plus d'informations, consultez le guide de programmation de votre contrôleur.
s/o	i_uiAssemblyInstanceOutput	UINT	Numéro de l'instance d'assemblage de sortie. Pour plus d'informations, consultez le guide de programmation de votre contrôleur.
s/o	i_uiAssemblyInstanceOutputSize	UINT	Taille de l'instance d'assemblage de sortie. Pour plus d'informations, consultez le guide de programmation de votre contrôleur.
<p>s/o signifie qu'aucun mappage %MW n'est prédéfini pour cette variable système. Non utilisée signifie que la variable n'est pas gérée par le système et que si sa valeur est différente de zéro, elle doit être considérée comme variable parasite.</p>			

%MW	Nom de la variable	Type	Commentaire
s/o	i_uiETHIP_ConnectionTimeouts	UINT	Nombre d'expirations de connexion. Réinitialisation lors de la mise sous tension ou avec la commande de réinitialisation ETH_W.q_wResetCounter.
s/o	i_ucEipRunIdle	ETH_R_RUN_IDLE <i>(voir page 83)</i>	Drapeau fonctionnement (valeur = 1) / attente (valeur = 0) pour la connexion EtherNet/IP classe 1.
s/o	i_byMasterIpTimeouts	REAL	Compteur d'événements de dépassement de délai TCP maître Ethernet Modbus. Réinitialisation lors de la mise sous tension ou avec la commande de réinitialisation ETH_W.q_wResetCounter.
s/o	i_byMasterIpLost	BYTE	Etat de la liaison du maître Modbus TCP Ethernet : 0 = liaison OK, 1 = liaison perdue.
s/o	i_wPortAIpStatus	ETH_R_PORT_IP_STATUS <i>(voir page 80)</i>	Etat de la pile du port TCP/IP Ethernet
s/o	i_byIPAddress_If2	ARRAY[0..3] OF BYTE	Non utilisé.
s/o	i_bySubNetMask_If2	ARRAY[0..3] OF BYTE	Non utilisé.
s/o	i_byGateway_If2	ARRAY[0..3] OF BYTE	Non utilisé.
s/o	i_byMACAddress_If2	ARRAY[0..5] OF BYTE	Non utilisé.
s/o	i_sDeviceName_If2	STRING(15)	Non utilisé.
s/o	i_wIpMode_If2	ETH_R_IP_MODE <i>(voir page 78)</i>	Non utilisé.
s/o	i_wPortALinkStatus_If2	ETH_R_PORT_LINK_STATUS <i>(voir page 81)</i>	Non utilisé.
s/o	i_wPortASpeed_If2	ETH_R_PORT_SPEED <i>(voir page 82)</i>	Non utilisé.
s/o	i_wPortADuplexStatus_If2	ETH_R_PORT_DUPLEX_STATUS <i>(voir page 79)</i>	Non utilisé.
s/o	i_wPortAIpStatus_If2	ETH_R_PORT_IP_STATUS <i>(voir page 80)</i>	Non utilisé.
<p>s/o signifie qu'aucun mappage %MW n'est prédéfini pour cette variable système. Non utilisée signifie que la variable n'est pas gérée par le système et que si sa valeur est différente de zéro, elle doit être considérée comme variable parasite.</p>			

NOTE : s/o signifie qu'aucun mappage %MW n'est prédéfini pour cette variable système.

ETH_W : variables système en lecture/écriture du port Ethernet

Structure de la variable

Le tableau suivant décrit les paramètres de la variable système ETH_W (type ETH_W_STRUCT) :

%MW	Nom de la variable	Type	Commentaire
s/o	q_wResetCounter	WORD	Le passage de 0 à 1 réinitialise tous les compteurs ETH_R. Pour réinitialiser à nouveau les compteurs, il est nécessaire de mettre ce registre à 0 de sorte qu'un nouveau passage de 0 à 1 puisse intervenir.

NOTE : s/o signifie qu'aucun mappage %MW n'est prédéfini pour cette variable système.

Sous-chapitre 1.5

Structure TM3_MODULE_R

TM3_MODULE_R[0...13] : Variables système en lecture seule des modules TM3

Introduction

TM3_MODULE_R est un tableau de 14 variables de type TM3_MODULE_R_STRUCT. Chaque élément du tableau renvoie des variables système de diagnostic pour le module d'extension TM3 correspondant.

Pour le Modicon M241 Logic Controller :

- TM3_MODULE_R[0] désigne le module d'extension TM3 numéro 0
- ...
- TM3_MODULE_R[13] désigne le module d'extension TM3 numéro 13

Structure de la variable

Le tableau suivant décrit les paramètres de la variable système TM3_MODULE_R[0...13] :

%MW	Nom de la variable	Type	Commentaire
s/o	i_wProductID	WORD	ID du module d'extension TM3.
s/o	i_wModuleState	TM3_MODULE_STATE (voir page 87)	Décrit l'état du module TM3.

NOTE : s/o signifie qu'aucun mappage %MW n'est prédéfini pour cette variable système.

Sous-chapitre 1.6

Structure PROFIBUS_R

PROFIBUS_R : Variables système en lecture seule de PROFIBUS

Structure de la variable

Le tableau suivant décrit les paramètres de la variable système PROFIBUS_R (type PROFIBUS_R_STRUCT) :

%MW	Nom de la variable	Type	Commentaire
s/o	i_wPNOIdentifier	WORD	Code d'identification d'esclave.
s/o	i_wBusAdr	UINT	Adresse d'esclave PROFIBUS
s/o	i_CommState	UDINT	Valeur représentant l'état du module PROFIBUS : <ul style="list-style-type: none"> ● 0x00 : Inconnu ● 0x01 : Non configuré ● 0x02 : Arrêt ● 0x03 : Inactif ● 0x04 : Fonctionnement
s/o	i_CommError	UDINT	Code d'erreur de communication.
s/o	i_ErrorCount	UDINT	Compteur d'erreurs de communication.

NOTE : s/o signifie qu'aucun mappage %MW n'est prédéfini pour cette variable système.

Sous-chapitre 1.7

Structure CART_R

CART_R_STRUCT : Variables système en lecture seule des cartouches

Structure de la variable

Le tableau suivant décrit les paramètres de la variable système `CART_R_STRUCT` :

%MW	Nom de la variable	Type	Commentaire
s/o	<code>i_uiModuleId</code>	<code>CART_R_MODULE_ID</code> <i>(voir page 90)</i>	ID module
s/o	<code>i_uifirmwareVersion</code>	UINT	Version du micrologiciel
s/o	<code>i_udiCartState</code>	<code>CART_R_STATE</code> <i>(voir page 91)</i>	Etat de la cartouche

NOTE : s/o signifie qu'aucun mappage %MW n'est prédéfini pour cette variable système.

Chapitre 2

Fonctions système de M241

Présentation

Ce chapitre décrit les fonctions système disponibles dans la bibliothèque PLCSystem de M241.

Contenu de ce chapitre

Ce chapitre contient les sous-chapitres suivants :

Sous-chapitre	Sujet	Page
2.1	Fonctions de lecture de M241	40
2.2	Fonctions d'écriture de l'automate M241	47
2.3	Fonctions utilisateur de M241	51
2.4	Fonctions de lecture TM3	57

Sous-chapitre 2.1

Fonctions de lecture de M241

Présentation

Cette section décrit les fonctions de lecture de la bibliothèque PLCSystem de M241.

Contenu de ce sous-chapitre

Ce sous-chapitre contient les sujets suivants :

Sujet	Page
GetImmediateFastInput : Lit l'entrée d'une E/S experte intégrée	41
GetRtc : Obtenir l'horodatage	42
IsFirstMastColdCycle : indique si le cycle est le premier cycle MAST après un démarrage à froid	43
IsFirstMastCycle : indique si le cycle est le premier cycle MAST	44
IsFirstMastWarmCycle : indique si le cycle est le premier cycle MAST après un démarrage à chaud	46

GetImmediateFastInput : Lit l'entrée d'une E/S experte intégrée

Description de la fonction

Cette fonction renvoie la valeur physique actuelle de l'entrée, qui peut différer de la valeur logique actuelle de cette entrée. La valeur est immédiatement lue sur le matériel lors de l'appel de la fonction. Seules les entrées I0 à I7 sont accessibles à l'aide de cette fonction.

Représentation graphique



Représentation en langage IL et ST

Pour voir la représentation générale en langage IL ou ST, reportez-vous au chapitre *Représentation des fonctions et blocs fonction* (voir page 97).

Description des variables d'E/S

Le tableau suivant décrit les variables d'entrée :

Entrée	Type	Commentaire
Block	INT	Non utilisé.
Input	INT	Index de l'entrée à lire, de 0 à 7.

Le tableau suivant décrit la variable de sortie :

Sortie	Type	Commentaire
GetImmediateFastInput	BOOL	Valeur de l'entrée <Input> – FALSE/TRUE.

Le tableau suivant décrit les variables d'entrée/sortie :

Entrée/Sortie	Type	Commentaire
Erreur	BOOL	FALSE = opération correcte. TRUE = opération en erreur, la fonction renvoie une valeur qui n'est pas valide.
ErrID	IMMEDIATE_ERR_TYPE (voir page 93)	Code de l'erreur d'opération détectée lorsque Error a la valeur TRUE.

GetRtc : Obtenir l'horodatage

Description de la fonction

Cette fonction renvoie l'horodatage en secondes au format UNIX (nombre de secondes écoulées depuis le 1/1/1970 à minuit (UTC)).

Représentation graphique



Représentation en langage IL et ST

Pour voir la représentation générale en langage IL ou ST, reportez-vous au chapitre *Représentation des fonctions et blocs fonction* (voir page 97).

Description des variables d'E/S

Le tableau suivant décrit la variable d'entrée/sortie :

Sortie	Type	Commentaire
GetRtc	DINT	Horodatage actuel en secondes au format UNIX.

Exemple

L'exemple suivant montre comment obtenir la valeur d'horodatage :

```
VAR
    MyRTC : DINT := 0;
END_VAR
MyRTC := GetRtc();
```

IsFirstMastColdCycle : indique si le cycle est le premier cycle MAST après un démarrage à froid

Description de la fonction

Cette fonction renvoie TRUE au cours du premier cycle MAST après un démarrage à froid (premier cycle après téléchargement ou réinitialisation à froid).

Représentation graphique



Représentation en IL et en ST

Pour voir la représentation générale en langage IL ou ST, reportez-vous au chapitre *Représentation des fonctions et blocs fonction* (voir page 97).

Description des variables d'E/S

Ce tableau décrit la variable de sortie :

Sortie	Type	Commentaire
IsFirstMastColdCycle	BOOL	TRUE au cours du premier cycle de la tâche MAST après un démarrage à froid.

Exemple

Reportez-vous à la description de la fonction `IsFirstMastCycle` (voir page 44).

IsFirstMastCycle : indique si le cycle est le premier cycle MAST

Description de la fonction

Cette fonction renvoie TRUE lors du premier cycle MAST après un démarrage.

Représentation graphique



Représentation en IL et en ST

Pour voir la représentation générale en langage IL ou ST, reportez-vous au chapitre *Représentation des fonctions et blocs fonction* (voir page 97).

Description des variables d'E/S

Sortie	Type	Commentaire
IsFirstMastCycle	BOOL	TRUE lors du premier cycle de la tâche MAST après un démarrage.

Exemple

Cet exemple décrit les trois fonctions `IsFirstMastCycle`, `IsFirstMastColdCycle` et `IsFirstMastWarmCycle` utilisées ensemble.

Utilisez cet exemple dans la tâche MAST. Sinon, il peut s'exécuter plusieurs fois ou jamais (une tâche supplémentaire peut être appelée plusieurs fois ou éventuellement aucune fois pendant un cycle de tâche MAST) :

```
VAR MyIsFirstMastCycle : BOOL; MyIsFirstMastWarmCycle : BOOL; MyIsFirst-
MastColdCycle : BOOL; END_VAR

MyIsFirstMastWarmCycle := IsFirstMastWarmCycle(); MyIsFirstMastColdCycle
:= IsFirstMastColdCycle(); MyIsFirstMastCycle := IsFirstMastCycle();

IF (MyIsFirstMastWarmCycle) THEN

(*Il s'agit du premier cycle MAST après un démarrage à chaud : toutes les
variables prennent sur leurs valeurs d'initialisation, à l'exception des
variables conservées.*)

(*=> initialisez les variables nécessaires pour que votre application
s'exécute comme prévu dans ce cas*)

END_IF;
```

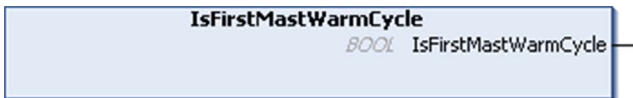
```
IF (MyIsFirstMastColdCycle) THEN
(*Il s'agit du premier cycle MAST après un démarrage à froid : toutes les
variables prennent sur leurs valeurs d'initialisation, y compris les
variables conservées.*)
(*=> initialisez les variables nécessaires pour que votre application
s'exécute comme prévu dans ce cas*)
END_IF;
IF (MyIsFirstMastCycle) THEN
(*Il s'agit du premier cycle MAST après un démarrage, c'est-à-dire après
un démarrage à chaud ou à froid ou l'exécution de commandes STOP/RUN*)
(*=> initialisez les variables nécessaires pour que votre application
s'exécute comme prévu dans ce cas*)
END_IF;
```

IsFirstMastWarmCycle : indique si le cycle est le premier cycle MAST après un démarrage à chaud

Description de la fonction

Cette fonction renvoie TRUE lors du premier cycle MAST après un démarrage à chaud.

Représentation graphique



Représentation en IL et en ST

Pour voir la représentation générale en langage IL ou ST, reportez-vous au chapitre *Représentation des fonctions et blocs fonction* (voir page 97).

Description des variables d'E/S

Ce tableau décrit la variable de sortie :

Sortie	Type	Commentaire
IsFirstMastWarmCycle	BOOL	TRUE au cours du premier cycle de la tâche MAST après un démarrage à chaud.

Exemple

Reportez-vous à la description de la fonction IsFirstMastCycle (voir page 44).

Sous-chapitre 2.2

Fonctions d'écriture de l'automate M241

Vue d'ensemble

Cette section décrit les fonctions d'écriture de la bibliothèque PLCSystem de l'automate M241.

Contenu de ce sous-chapitre

Ce sous-chapitre contient les sujets suivants :

Sujet	Page
PhysicalWriteFastOutputs : Ecrit la sortie rapide d'une E/S experte intégrée	48
SetRTCDrift : Définir la valeur de compensation de l'horodateur	49

PhysicalWriteFastOutputs : Écrit la sortie rapide d'une E/S experte intégrée

Description de la fonction

Cette fonction écrit un état physique dans les sorties Q0 à Q3 au moment où elle est appelée.

Représentation graphique



Représentation en IL et en ST

Pour voir la représentation générale en langage IL ou ST, reportez-vous au chapitre *Représentation des fonctions et blocs fonction* (voir page 97).

Description des variables d'E/S

Le tableau suivant décrit les variables d'entrée :

Entrée	Type	Commentaire
Q0Value	BOOL	La valeur est demandée pour la sortie 0.
Q1Value	BOOL	La valeur est demandée pour la sortie 1.
Q2Value	BOOL	La valeur est demandée pour la sortie 2.
Q3Value	BOOL	La valeur est demandée pour la sortie 3.

Le tableau suivant décrit les variables de sortie :

Sortie	Type	Commentaire
PhysicalWriteFastOutputs	WORD	Valeur de sortie de la fonction.

NOTE : Seuls les 4 premiers bits de la valeur renvoyée sont significatifs et utilisés comme champ de bits pour indiquer si la sortie est écrite.

NOTE : Si le bit correspondant à la sortie est 1, l'écriture de cette sortie a réussi.

NOTE : Si le bit correspondant à la sortie est 0, l'écriture n'a pas été effectuée car cette sortie est déjà utilisée par une fonction experte.

NOTE : Si le bit correspondant à la sortie est 0b1111, l'écriture des 4 sorties a réussi.

NOTE : Si le bit correspondant à la sortie est 0b1110, Q0 n'est pas écrite car elle est utilisée par un générateur de fréquence.

SetRTCDrift : Définir la valeur de compensation de l'horodateur

Description de la fonction

Cette fonction accélère ou ralentit la fréquence de l'horodateur afin de donner la main à l'application pour compenser l'horodateur en fonction de l'environnement de fonctionnement (température, ...). La valeur de compensation est donnée en secondes par semaine. Elle peut être positive (accélération) ou négative (ralentissement).

NOTE : La fonction `SetRTCDrift` doit être appelée une seule fois. Chaque nouvel appel remplace la valeur de compensation par une nouvelle. Cette valeur est conservée dans le matériel du contrôleur tant que l'horodateur est alimenté par le secteur ou la batterie. En cas de suppression des deux sources d'alimentation, la valeur de compensation de l'horodateur est perdue.

Représentation graphique



Représentation en IL et en ST

Pour voir la représentation générale en langage IL ou ST, reportez-vous au chapitre *Représentation des fonctions et blocs fonction* (voir page 97).

Description des variables d'E/S

Le tableau suivant décrit les paramètres d'entrée :

Entrées	Type	Commentaire
RTCDrift	SINT (-29..29)	Correction en secondes par semaine (-29 ... +29).

NOTE : Les paramètres `Day`, `Hour` et `Minute` sont utilisés uniquement pour assurer la compatibilité descendante.

NOTE : Si la valeur entrée pour `RtcDrift` dépasse la limite, le micrologiciel du contrôleur la remplace par la valeur maximale.

Le tableau suivant décrit la variable de sortie :

Sortie	Type	Commentaire
SetRTCDrift	RTCSETDRIFT_ERROR (voir page 94)	Renvoie <code>RTC_OK</code> (00 hex) si la commande est correcte ou renvoie le code d'identification de l'erreur détectée.

Exemple

Dans cet exemple, la fonction est appelée une seule fois pendant le premier cycle de tâche MAST. Elle accélère l'horodateur de 4 secondes par semaine (18 secondes par mois).

```
VAR
    MyRTCDrift : SINT (-29..29) := 0;
    MyDay : DAY_OF_WEEK;
    MyHour : HOUR;
    MyMinute : MINUTE;
END_VAR
IF IsFirstMastCycle() THEN
    MyRTCDrift := 4;
    MyDay := 0;
    MyHour := 0;
    MyMinute := 0;
    SetRTCDrift(MyRTCDrift, MyDay, MyHour, MyMinute);
END_IF
```

Sous-chapitre 2.3

Fonctions utilisateur de M241

Présentation

Cette section décrit les fonctions `DataFileCopy` et `ExecuteScript` disponibles dans la bibliothèque PLCSystem de M241.

Contenu de ce sous-chapitre

Ce sous-chapitre contient les sujets suivants :

Sujet	Page
<code>DataFileCopy</code> : Commandes de copie de fichier	52
<code>ExecuteScript</code> : Commandes de script	55

DataFileCopy : Commandes de copie de fichier

Description de la fonction

Cette fonction copie les données en mémoire dans un fichier, et inversement. Le fichier réside dans le système de fichiers interne ou dans un système de fichiers externe (carte SD).

Le bloc fonction `DataFileCopy` peut effectuer les opérations suivantes :

- lire les données d'un fichier formaté ;
- copier les données de la mémoire tampon dans un fichier formaté. Pour plus d'informations, reportez-vous à la section Flash Memory Organization (*voir Modicon M241 Logic Controller, Guide de programmation*).

Représentation graphique



Représentation en IL et en ST

Pour voir la représentation générale en langage IL ou ST, reportez-vous au chapitre *Représentation des fonctions et blocs fonction* (*voir page 97*).

Description des variables d'E/S

Le tableau suivant décrit les variables d'entrée :

Entrée	Type	Commentaire
<code>xExecute</code>	BOOL	Lors d'un front montant, lance l'exécution du bloc fonction. Sur le front descendant, réinitialise les sorties du bloc fonction à la fin de son exécution.
<code>sFileName</code>	STRING	Nom de fichier sans l'extension (l'extension <code>.DTA</code> est ajoutée automatiquement). N'utilisez que les caractères alphanumériques (a à z, A à Z et 0 à 9).
<code>xRead</code>	BOOL	TRUE : copie les données du fichier dans la mémoire. FALSE : copie les données de la mémoire dans le fichier.
<code>xSecure</code>	BOOL	TRUE : l'adresse MAC est toujours stockée dans le fichier. Seul un contrôleur ayant la même adresse MAC peut lire le contenu du fichier. FALSE : un autre contrôleur ayant le même type de mémoire peut lire le contenu du fichier.

Entrée	Type	Commentaire
iLocation	INT	0 : le fichier réside dans le répertoire /usr/Dta du système de fichiers interne. 1 : le fichier réside dans le répertoire /usr/Dta du système de fichiers externe (carte SD).
uiSize	UINT	Indique la taille en octets. Le maximum est 65534 octets. Seules les adresses de variables conformes à la norme CEI 6113-1 (variables, tableaux, structures) sont autorisées. Par exemple : Variable : int; uiSize := SIZEOF (Variable);
dwAdd	DWORD	Indique l'adresse dans la mémoire. Seules les adresses de variables conformes à la norme CEI 6113-1 (variables, tableaux, structures) sont autorisées. Par exemple : Variable : int; dwAdd := ADR (Variable);

AVERTISSEMENT

COMPORTEMENT INATTENDU DE L'EQUIPEMENT

Vérifiez que la taille de la mémoire et le type du fichier sont corrects avant de copier le fichier dans la mémoire.

Le non-respect de ces instructions peut provoquer la mort, des blessures graves ou des dommages matériels.

Le tableau suivant décrit les variables de sortie :

Sortie	Type	Commentaire
xDone	BOOL	TRUE = indique que l'action a abouti.
xBusy	BOOL	TRUE = indique que le bloc fonction s'exécute.
xError	BOOL	TRUE = indique qu'une erreur est détectée et que le bloc fonction a annulé l'action.
eError	DataFileCopyError (voir page 73)	Indique le type de l'erreur détectée lors de la copie du fichier de données.

NOTE : si vous écrivez dans une variable mémoire au sein de la zone d'écriture du fichier, une erreur CRC est détectée.

Exemple

L'exemple suivant montre comment utiliser les commandes de copie de fichier :

```
VAR
LocalArray : ARRAY [0..29] OF BYTE;
myFileName: STRING := 'exportfile';
EXEC_FLAG: BOOL;
DataFileCopy: DataFileCopy;
END_VAR
DataFileCopy(
  xExecute:= EXEC_FLAG,
  sFileName:= myFileName,
  xRead:= FALSE,
  xSecure:= FALSE,
  iLocation:= DFCL_INTERNAL,
  dwSize:= SIZEOF(LocalArray),
  dwAdd:= ADR(LocalArray),
  xDone=> ,
  xBusy=> ,
  xError=> ,
  eError=> );
```

ExecuteScript : Commandes de script

Description de la fonction

Cette fonction peut exécuter les commandes de script de carte SD suivantes :

- Download
- Upload
- SetNodeName
- Delete
- Reboot

Respectez la syntaxe employée dans le script USB pour exécuter ces commandes (majuscules/minuscules). Reportez-vous à Génération de scripts et de fichiers avec stockage de masse sur carte SD (*voir Modicon M241 Logic Controller, Guide de programmation*).

Représentation graphique



Représentation en IL et en ST

Pour voir la représentation générale en langage IL ou ST, reportez-vous au chapitre *Représentation des fonctions et blocs fonction* (*voir page 97*).

Description des variables d'E/S

Le tableau suivant décrit les variables d'entrée :

Entrée	Type	Commentaire
xExecute	BOOL	Lors d'un <i>rising edge</i> , lance l'exécution du bloc fonction. Lors d'un <i>falling edge</i> , réinitialise les sorties du bloc fonction à la fin de son exécution.
sCmd	STRING	Syntaxe de la commande de script sur carte SD. L'exécution simultanée de commandes n'est pas autorisée : si une commande est exécutée par un autre bloc fonction ou un script de carte SD, le bloc fonction ne l'exécute pas immédiatement mais la met en file d'attente. NOTE : Un script de carte SD exécuté à partir d'une carte SD est considéré comme en cours d'exécution jusqu'à ce que la carte soit retirée.

Le tableau suivant décrit les variables de sortie :

Sortie	Type	Commentaire
xDone	BOOL	TRUE = indique que l'action a abouti.
xBusy	BOOL	TRUE = le bloc fonction est en cours d'exécution.
xError	BOOL	TRUE = indique la détection d'une erreur et le bloc fonction abandonne l'action.
eError	ExecuteScriptError (voir page 75)	Indique le type d'erreur détecté dans le script d'exécution.

Exemple

L'exemple suivant montre comment exécuter une commande de script :

```
VAR
EXEC_FLAG: BOOL;
ExecuteScript: ExecuteScript;
END_VAR
ExecuteScript(
xExecute:= EXEC_FLAG,
sCmd:= `Upload "/usr/Syslog/*"`,
xDone=> ,
xBusy=> ,
xError=> ,
eError=> );
```


Sous-chapitre 2.4

Fonctions de lecture TM3

Présentation

Cette section décrit les fonctions de lecture TM3 incluses dans la bibliothèque PLCSystem du M241.

Contenu de ce sous-chapitre

Ce sous-chapitre contient les sujets suivants :

Sujet	Page
TM3_GetModuleBusStatus : Obtient l'état du bus des modules TM3	58
TM3_GetModuleInternalStatus : Obtient l'état interne des modules TM3	59

TM3_GetModuleBusStatus : Obtient l'état du bus des modules TM3

Description de la fonction

Cette fonction renvoie le statut de bus du module. L'index du module est fourni en tant que paramètre d'entrée.

Représentation graphique



Représentation en IL et en ST

Pour voir la représentation générale en langage IL ou ST, reportez-vous au chapitre *Représentation des fonctions et blocs fonction* (voir page 97).

Description des variables d'E/S

Le tableau suivant décrit la variable d'entrée :

Entrée	Type	Commentaire
ModuleIndex	BYTE	Index du module (0 pour la première extension, 1 pour la deuxième, etc.).

Le tableau suivant décrit les variables de sortie :

Sortie	Type	Commentaire
TM3_GetModuleBusStatus	TM3_ERR_CODE (voir page 85)	Renvoie TM3_OK (00 hex) si la commande est correcte ou renvoie le code d'identification de l'erreur détectée.

TM3_GetModuleInternalStatus : Obtient l'état interne des modules TM3

Description de la fonction

Cette fonction remplit la mémoire tampon `pStatusBuffer` avec la table d'états du module `ModuleIndex`.

Représentation graphique



Représentation en langage IL et ST

Pour voir la représentation générale en langage IL ou ST, reportez-vous au chapitre *Représentation des fonctions et blocs fonction* (voir page 97).

Description des variables d'E/S

⚠ AVERTISSEMENT
COMPORTEMENT INATTENDU DE L'EQUIPEMENT
Vérifiez que le paramètre <code>pStatusBuffer</code> est alloué.
Le non-respect de ces instructions peut provoquer la mort, des blessures graves ou des dommages matériels.

Le tableau suivant décrit les variables d'entrée :

Entrée	Type	Commentaire
<code>ModuleIndex</code>	BYTE	Index du module (0 pour la première extension, 1 pour la deuxième, etc.).
<code>StatusOffset</code>	BYTE	Décalage du premier état à lire dans la table d'états.
<code>StatusSize</code>	BYTE	Nombre d'octets à lire dans la table d'états.
<code>pStatusBuffer</code>	POINTER TO BYTE	Mémoire tampon contenant la table d'états lus.

Le tableau suivant décrit la variable de sortie :

Sortie	Type	Commentaire
TM3_GetModuleInternalStatus	TM3_ERR_CODE <i>(voir page 85)</i>	Renvoie TM3_OK (00 hex) si la commande est correcte, sinon renvoie le code d'identification de l'erreur détectée.

Exemple

L'exemple suivant montre comment obtenir l'état interne des modules :

```
VAR  
AMM3HT_Channel1_Input_Status: BYTE;  
END_VAR  
TM3_GetModuleInternalStatus(0, 1, 1, ADR(AMM3HT_Channel1_Input_Status));
```

Chapitre 3

Types de données de la bibliothèque PLCSystem M241

Présentation

Ce chapitre décrit les types de données de la bibliothèque PLCSystem de M241.

Deux types de données sont disponibles :

- Les types de données de variable système sont utilisés par les variables système (*voir page 13*) (PLC_R, PLC_W,...) de la bibliothèque PLCSystem de M241.
- Les types de données de fonction système sont utilisés par les fonctions système (*voir page 39*) de lecture/écriture de la bibliothèque PLCSystem de M241.

Contenu de ce chapitre

Ce chapitre contient les sous-chapitres suivants :

Sous-chapitre	Sujet	Page
3.1	Types de données des variables système PLC_R/W	62
3.2	Types de données des variables système DataFileCopy	72
3.3	Types de données des variables système ExecScript	75
3.4	Types de données des variables système ETH_R/W	76
3.5	Types de données des variables système TM3_MODULE_R	84
3.6	Types de données des variables système des cartouches	88
3.7	Types de données des fonctions système	92

Sous-chapitre 3.1

Types de données des variables système PLC_R/W

Présentation

Cette section répertorie et décrit les types de données de variable système, inclus dans les structures PLC_R et PLC_W.

Contenu de ce sous-chapitre

Ce sous-chapitre contient les sujets suivants :

Sujet	Page
PLC_R_APPLICATION_ERROR : Codes d'état des erreurs de l'application détectées	63
PLC_R_BOOT_PROJECT_STATUS : codes d'état du projet de démarrage	64
types de données PLC_R_IO_STATUS : codes d'état des E/S	65
PLC_R_SDCARD_STATUS : Codes d'état de l'emplacement de carte SD	66
PLC_R_STATUS : codes d'état du contrôleur	67
PLC_R_STOP_CAUSE : Codes des causes de transition de l'état RUN à un autre	68
PLC_R_TERMINAL_PORT_STATUS : codes d'état de la connexion du port de programmation	69
PLC_R_TM3_BUS_STATE : Codes d'état de bus TM3	70
PLC_W_COMMAND : codes de commande de contrôle	71

PLC_R_APPLICATION_ERROR : Codes d'état des erreurs de l'application détectées

Description du type énumération

Le type de données énumération PLC_R_APPLICATION_ERROR contient les valeurs suivantes :

Enumérateur	Valeur	Commentaire
PLC_R_APP_ERR_UNKNOWN	FFFF hex	Erreur indéfinie détectée.
PLC_R_APP_ERR_NOEXCEPTION	0000 hex	Aucune erreur détectée.
PLC_R_APP_ERR_WATCHDOG	0010 hex	Le chien de garde de l'application de la tâche a expiré.
PLC_R_APP_ERR_HARDWAREWATCHDOG	0011 hex	Le chien de garde matériel a expiré.
PLC_R_APP_ERR_IO_CONFIG_ERROR	0012 hex	Paramètres de configuration d'E/S incorrects détectés.
PLC_R_APP_ERR_UNRESOLVED_EXTREFS	0018 hex	Fonctions indéfinies détectées.
PLC_R_APP_ERR_IEC_TASK_CONFIG_ERROR	0025 hex	Paramètres de configuration de tâche incorrects détectés.
PLC_R_APP_ERR_ILLEGAL_INSTRUCTION	0050 hex	Instruction indéfinie détectée.
PLC_R_APP_ERR_ACCESS_VIOLATION	0051 hex	Tentative d'accès à la zone mémoire réservée.
PLC_R_APP_ERR_DIVIDE_BY_ZERO	0102 hex	Division d'un entier par 0 détectée.
PLC_R_APP_ERR_PROCESSORLOAD_WATCHDOG	0105 hex	Processeur surchargé par les tâches de l'application.
PLC_R_APP_ERR_DIVIDE_REAL_BY_ZERO	0152 hex	Division d'un réel par 0 détectée.
PLC_R_APP_ERR_EXPIO_EVENTS_COUNT_EXCEEDED	4E20 hex	Trop d'événements sur les E/S expertes sont détectés. Réduisez le nombre de tâches d'événement.
PLC_R_APP_ERR_APPLICATION_VERSION_MISMATCH	4E21 hex	Différence détectée dans la version de l'application.

PLC_R_BOOT_PROJECT_STATUS : codes d'état du projet de démarrage

Description du type énuméré

Le type de données énuméré PLC_R_BOOT_PROJECT_STATUS contient les valeurs suivantes :

Enumérateur	Valeur	Commentaire
PLC_R_NO_BOOT_PROJECT	0000 hex	Le projet de démarrage n'existe pas dans la mémoire Flash.
PLC_R_BOOT_PROJECT_CREATION_IN_PROGRESS	0001 hex	Le projet de démarrage est en cours de création.
PLC_R_DIFFERENT_BOOT_PROJECT	0002 hex	Le projet de démarrage dans la mémoire Flash est différent du projet chargé dans la RAM.
PLC_R_VALID_BOOT_PROJECT	FFFF hex	Le projet de démarrage dans la mémoire Flash est identique au projet chargé dans la RAM.

types de données PLC_R_IO_STATUS : codes d'état des E/S

Description du type énuméré

Le type de données énuméré PLC_R_IO_STATUS contient les valeurs suivantes :

Enumérateur	Valeur	Commentaire
PLC_R_IO_OK	FFFF hex	Les entrées/sorties sont opérationnelles.
PLC_R_IO_NO_INIT	0001 hex	Les entrées/sorties ne sont pas initialisées.
PLC_R_IO_CONF_FAULT	0002 hex	Paramètres de configuration d'E/S incorrects détectés.
PLC_R_IO_SHORTCUT_FAULT	0003 hex	Court-circuit des entrées/sorties détecté.
PLC_R_IO_POWER_SUPPLY_FAULT	0004 hex	Erreur d'alimentation des E/S détectée.

PLC_R_SDCARD_STATUS : Codes d'état de l'emplacement de carte SD

Description du type énumération

Le type de données énumération PLC_R_SDCARD_STATUS contient les valeurs suivantes :

Enumérateur	Valeur	Commentaire
NO_SDCARD	0000 hex	Aucune carte SD n'est détectée dans l'emplacement ou l'emplacement n'est pas connecté.
SDCARD_READONLY	0001 hex	La carte SD est en mode de lecture seule.
SDCARD_READWRITE	0002 hex	La carte SD est en mode de lecture/écriture.
SDCARD_ERROR	0003 hex	Erreur détectée sur la carte SD.

PLC_R_STATUS : codes d'état du contrôleur

Description du type énuméré

Le type de données énuméré PLC_R_STATUS contient les valeurs suivantes :

Enumérateur	Valeur	Commentaire
PLC_R_EMPTY	0000 hex	Le contrôleur ne contient aucune application.
PLC_R_STOPPED	0001 hex	Le contrôleur est arrêté.
PLC_R_RUNNING	0002 hex	Le contrôleur fonctionne.
PLC_R_HALT	0004 hex	Le contrôleur est à l'état HALT. (Reportez-vous au diagramme d'état du contrôleur dans le <i>guide de programmation</i> de votre contrôleur.)
PLC_R_BREAKPOINT	0008 hex	Le contrôleur s'est interrompu au point d'arrêt.

PLC_R_STOP_CAUSE : Codes des causes de transition de l'état RUN à un autre

Description du type énumération

Le type de données d'énumération PLC_R_STOP_CAUSE contient les valeurs suivantes :

Enumérateur	Valeur	Commentaire
PLC_R_STOP_REASON_UNKNOWN	00 hex	La valeur initiale ou la cause de l'arrêt ne sont pas définies.
PLC_R_STOP_REASON_HW_WATCHDOG	01 hex	Arrêté suite au timeout du chien de garde matériel
PLC_R_STOP_REASON_RESET	02 hex	Arrêté suite à une réinitialisation.
PLC_R_STOP_REASON_EXCEPTION	03 hex	Arrêté suite à une exception.
PLC_R_STOP_REASON_USER	04 hex	Arrêté suite à une requête de l'utilisateur.
PLC_R_STOP_REASON_IECPROGRAM	05 hex	Arrêté suite à une requête de commande de programme (par exemple, commande de contrôle avec paramètre PLC_W.q_wPLCControl:=PLC_W_COMMAND.PLC_W_STOP;).
PLC_R_STOP_REASON_DELETE	06 hex	Arrêté suite à une commande de suppression d'application.
PLC_R_STOP_REASON_DEBUGGING	07 hex	Arrêté suite au passage en mode de débogage.
PLC_R_STOP_FROM_NETWORK_REQUEST	0A hex	Arrêté suite à une requête du réseau, de la carte SD ou de la commande PLC_W command.
PLC_R_STOP_FROM_INPUT	0B hex	Arrêt requis par une entrée du contrôleur.
PLC_R_STOP_FROM_RUN_STOP_SWITCH	0C hex	Arrêt demandé par le commutateur du contrôleur.
PLC_R_STOP_REASON_RETAIN_MISMATCH	0D hex	Arrêté suite à un échec du test de vérification du contexte lors du redémarrage.
PLC_R_STOP_REASON_BOOT_APPLI_MISMATCH	0E hex	Arrêté suite à un échec de la comparaison entre l'application de démarrage et celle qui était en mémoire avant le redémarrage.
PLC_R_STOP_REASON_POWERFAIL	0F hex	Arrêté suite à une coupure de courant.

Pour plus d'informations sur les raisons de l'arrêt du contrôleur, consultez la section Description des états du contrôleur.

PLC_R_TERMINAL_PORT_STATUS : codes d'état de la connexion du port de programmation

Description du type énumération

Le type de données énumération PLC_R_TERMINAL_PORT_STATUS contient les valeurs suivantes :

Enumérateur	Valeur	Commentaire
TERMINAL_NOT_CONNECTED	00 hex	Aucun PC n'est connecté au port de programmation.
TERMINAL_CONNECTION_IN_PROGRESS	01 hex	Connexion en cours.
TERMINAL_CONNECTED	02 hex	PC connecté au port de programmation.
TERMINAL_ERROR	0F hex	Erreur détectée lors de la connexion.

PLC_R_TM3_BUS_STATE : Codes d'état de bus TM3

Description du type énumération

Le type de données énumération PLC_R_TM3_BUS_STATE contient les valeurs suivantes :

Enumérateur	Valeur	Commentaire
TM3_CONF_ERROR	01 hex	Erreur détectée en raison d'une incohérence entre la configuration physique et la configuration dans SoMachine.
TM3_OK	03 hex	La configuration physique correspond à la configuration dans SoMachine.
TM3_POWER_SUPPLY_ERROR	04 hex	Erreur détectée dans l'alimentation.

PLC_W_COMMAND : codes de commande de contrôle

Description du type énuméré

Le type de données énuméré PLC_W_COMMAND contient les valeurs suivantes :

Enumérateur	Valeur	Commentaire
PLC_W_STOP	0001 hex	Commande d'arrêt du contrôleur.
PLC_W_RUN	0002 hex	Commande d'exécution du contrôleur.
PLC_W_RESET_COLD	0004 hex	Commande de lancement d'une réinitialisation à froid du contrôleur.
PLC_W_RESET_WARM	0008 hex	Commande de lancement d'une réinitialisation à chaud du contrôleur.

Sous-chapitre 3.2

Types de données des variables système DataFileCopy

Présentation

Cette section répertorie et décrit les types de données des variables système incluses dans les structures `DataFileCopy`.

Contenu de ce sous-chapitre

Ce sous-chapitre contient les sujets suivants :

Sujet	Page
DataFileCopyError : codes des erreurs détectées	73
DataFileCopyLocation : Codes d'emplacement	74

DataFileCopyError : codes des erreurs détectées

Description du type énumération

Le type de données énumération `DataFileCopyError` contient les valeurs suivantes :

Enumérateur	Valeur	Description
ERR_NO_ERR	00 hex	Aucune erreur détectée.
ERR_FILE_NOT_FOUND	01 hex	Fichier inexistant.
ERR_FILE_ACCESS_REFUSED	02 hex	Ouverture du fichier impossible.
ERR_INCORRECT_SIZE	03 hex	Requête d'une taille différente de celle indiquée dans le fichier.
ERR_CRC_ERR	04 hex	CRC incorrect. Le fichier est considéré comme endommagé.
ERR_INCORRECT_MAC	05 hex	Le contrôleur tentant de lire le fichier n'a pas la même adresse MAC que celle indiquée dans le fichier.

DataFileCopyLocation : Codes d'emplacement

Description du type énumération

Le type de données énumération `DataFileCopyLocation` contient les valeurs suivantes :

Enumérateur	Valeur	Description
DFCL_INTERNAL	00 hex	Le fichier de donnée d'extension DTA se trouve dans le répertoire <i>/usr/Dta</i> .
DFCL_EXTERNAL	01 hex	Le fichier de donnée d'extension DTA se trouve dans le répertoire <i>/sd0/usr/Dta</i> .
DFCL_TBD	02 hex	Non utilisé.

Sous-chapitre 3.3

Types de données des variables système ExecScript

ExecuteScriptError : codes des erreurs détectées

Description du type énumération

Le type de données énumération `ExecuteScriptError` contient les valeurs suivantes :

Enumérateur	Valeur	Description
<code>CMD_OK</code>	00 hex	Aucune erreur détectée.
<code>ERR_CMD_UNKNOWN</code>	01 hex	Commande non reconnue.
<code>ERR_SD_CARD_MISSING</code>	02 hex	Carte SD absente.
<code>ERR_SEE_FWLOG</code>	03 hex	Erreur détectée lors de l'exécution de la commande. Consultez le fichier <code>FwLog.txt</code> . Pour plus d'informations, reportez-vous à la section Type de fichier (<i>voir Modicon M241 Logic Controller, Guide de programmation</i>).
<code>ERR_ONLY_ONE_COMMAND_ALLOWED</code>	04 hex	Tentative d'exécution de plusieurs scripts simultanément.
<code>CMD_BEING_EXECUTED</code>	05 hex	Un script est déjà en cours.

Sous-chapitre 3.4

Types de données des variables système ETH_R/W

Présentation

Cette section répertorie et décrit les types de données de variable système, inclus dans les structures `ETH_R` et `ETH_W`.

Contenu de ce sous-chapitre

Ce sous-chapitre contient les sujets suivants :

Sujet	Page
ETH_R_FRAME_PROTOCOL : codes de protocole de transmission de trames	77
ETH_R_IP_MODE : codes sources d'adresse IP	78
ETH_R_PORT_DUPLEX_STATUS : codes du mode de transmission	79
ETH_R_PORT_IP_STATUS : codes d'état du port TCP/IP Ethernet	80
ETH_R_PORT_LINK_STATUS : codes d'état de la liaison de communication	81
ETH_R_PORT_SPEED : codes de la vitesse de communication des ports Ethernet	82
ETH_R_RUN_IDLE : codes d'état fonctionnement et attente Ethernet/IP	83

ETH_R_FRAME_PROTOCOL : codes de protocole de transmission de trames

Description du type énuméré

Le type de données énuméré `ETH_R_FRAME_PROTOCOL` contient les valeurs suivantes :

Enumérateur	Valeur	Commentaire
<code>ETH_R_802_3</code>	00 hex	Le protocole utilisé pour la transmission des trames est IEEE 802.3.
<code>ETH_R_ETHERNET_II</code>	01 hex	Le protocole utilisé pour la transmission des trames est Ethernet II.

ETH_R_IP_MODE : codes sources d'adresse IP

Description du type énuméré

Le type de données énuméré ETH_R_IP_MODE contient les valeurs suivantes :

Enumérateur	Valeur	Commentaire
ETH_R_STORED	00 hex	L'adresse IP stockée est utilisée.
ETH_R_BOOTP	01 hex	Le protocole Bootstrap est utilisé pour obtenir une adresse IP.
ETH_R_DHCP	02 hex	Le protocole DHCP est utilisé pour obtenir une adresse IP.
ETH_DEFAULT_IP	FF hex	L'adresse IP par défaut est utilisée.

ETH_R_PORT_DUPLEX_STATUS : codes du mode de transmission

Description du type énumération

Le type de données énumération ETH_R_PORT_DUPLEX_STATUS contient les valeurs suivantes :

Enumérateur	Valeur	Commentaire
ETH_R_PORT_HALF_DUPLEX	00 hex	Le mode de transmission en semi-duplex est utilisé.
ETH_R_FULL_DUPLEX	01 hex	Le mode de transmission en duplex intégral est utilisé.
ETH_R_PORT_NA_DUPLEX	03 hex	Le mode de transmission sans duplex est utilisé.

ETH_R_PORT_IP_STATUS : codes d'état du port TCP/IP Ethernet

Description du type énumération

Le type de données énumération ETH_R_PORT_IP_STATUS contient les valeurs suivantes :

Enumérateur	Valeur	Commentaire
WAIT_FOR_PARAMS	00 hex	Attente de paramètres.
WAIT_FOR_CONF	01 hex	Attente de configuration.
DATA_EXCHANGE	02 hex	Prêt pour l'échange de données.
ETH_ERROR	03 hex	Erreur détectée sur le port TCP/IP Ethernet (câble déconnecté, configuration non valide, etc.).
DUPLICATE_IP	04 hex	Adresse IP déjà utilisée par un autre équipement.

ETH_R_PORT_LINK_STATUS : codes d'état de la liaison de communication

Description du type énumération

Le type de données énumération ETH_R_PORT_LINK_STATUS contient les valeurs suivantes :

Enumérateur	Valeur	Commentaire
ETH_R_LINK_DOWN	00 hex	Liaison de communication non disponible pour un autre équipement.
ETH_R_LINK_UP	01 hex	Liaison de communication disponible pour un autre équipement.

ETH_R_PORT_SPEED : codes de la vitesse de communication des ports Ethernet

Description du type énumération

Le type de données énumération ETH_R_PORT_SPEED contient les valeurs suivantes :

Enumérateur	Valeur	Commentaire
ETH_R_SPEED_NA	0 déc	Le débit réseau est de 0 mégabit par seconde.
ETH_R_SPEED_10_MB	10 déc	Le débit réseau est de 10 mégabits par seconde.
ETH_R_100_MB	100 déc	Le débit réseau est de 100 mégabits par seconde.

ETH_R_RUN_IDLE : codes d'état fonctionnement et attente Ethernet/IP

Description du type énumération

Le type de données énumération `ETH_R_RUN_IDLE` contient les valeurs suivantes :

Enumérateur	Valeur	Commentaire
IDLE	00 hex	La connexion EtherNet/IP est en attente.
RUN	01 hex	La connexion EtherNet/IP est en fonctionnement.

Sous-chapitre 3.5

Types de données des variables système

TM3_MODULE_R

Présentation

Cette section répertorie et décrit les types de données pour les variables système de la structure TM3_MODULE_R.

Contenu de ce sous-chapitre

Ce sous-chapitre contient les sujets suivants :

Sujet	Page
TM3_ERR_CODE : Codes d'erreur des modules d'extension TM3	85
TM3_MODULE_R_ARRAY_TYPE : Type tableau de lecture des modules d'extension TM3	86
TM3_MODULE_STATE : Codes d'état des modules d'extension TM3	87

TM3_ERR_CODE : Codes d'erreur des modules d'extension TM3

Description du type énumération

Le type de données énumération TM3_ERR_CODE contient les valeurs suivantes :

Enumérateur	Valeur	Commentaire
TM3_NO_ERR	00 hex	Le dernier échange du bus avec le module d'extension a réussi.
TM3_ERR_FAILED	01 hex	Erreur détectée suite à l'échec du dernier échange du bus avec le module d'extension.
TM3_ERR_PARAMETER	02 hex	Erreur de paramètre détectée dans le dernier échange du bus avec le module.
TM3_ERR_COK	03 hex	Erreur matérielle temporaire ou permanente détectée sur l'un des modules d'extension TM3.
TM3_ERR_BUS	04 hex	Erreur de bus détectée dans le dernier échange du bus avec le module d'extension.

TM3_MODULE_R_ARRAY_TYPE : Type tableau de lecture des modules d'extension TM3

Description

TM3_MODULE_R_ARRAY_TYPE est un tableau de 0 à 13 variables de type TM.3_MODULE_R_STRUCT.

TM3_MODULE_STATE : Codes d'état des modules d'extension TM3

Description du type énumération

Le type de données énumération TM3_MODULE_STATE contient les valeurs suivantes :

Enumérateur	Valeur	Commentaire
TM3_EMPTY	00 hex	Absence de module.
TM3_CONF_ERROR	01 hex	Le module d'extension physique ne correspond pas à celui configuré dans SoMachine.
TM3_BUS_ERROR	02 hex	Erreur de bus détectée dans le dernier échange avec le module.
TM3_OK	03 hex	Le dernier échange du bus avec ce module a réussi.

Sous-chapitre 3.6

Types de données des variables système des cartouches

Présentation

Cette section répertorie et décrit les types de données des variables système comprises dans la structure `Cartridge`.

Contenu de ce sous-chapitre

Ce sous-chapitre contient les sujets suivants :

Sujet	Page
CART_R_ARRAY_TYPE : Type tableau de lecture des cartouches	89
CART_R_MODULE_ID : Identificateur de module de lecture de cartouche	90
CART_R_STATE : Etat de lecture des cartouches	91

CART_R_ARRAY_TYPE : Type tableau de lecture des cartouches

Description

CART_R_ARRAY_TYPE est un tableau de 0 ou 1 variables de type CART_R_STRUCT.

CART_R_MODULE_ID : Identificateur de module de lecture de cartouche

Description du type énumération

Le type de données énumération `CART_R_MODULE_ID` contient les valeurs suivantes :

Enumérateur	Valeur	Description
<code>CART_R_MODULE_ID</code>	40 hex	TMC4AI2
<code>CART_R_MODULE_ID</code>	41 hex	TMC4AQ2
<code>CART_R_MODULE_ID</code>	42 hex	TMC4TI2
<code>CART_R_MODULE_ID</code>	48 hex	TMC4HOIS01
<code>CART_R_MODULE_ID</code>	49 hex	TMC4PACK01
<code>CART_R_MODULE_ID</code>	FF hex	Aucune

CART_R_STATE : Etat de lecture des cartouches

Description du type énumération

Le type de données énumération `CART_R_STATE` contient les valeurs suivantes :

Enumérateur	Valeur	Commentaire
<code>CONFIGURED</code>	00 hex	La cartouche est configurée.
<code>INITIALIZED_NOT_CONFIGURED</code>	01 hex	La cartouche est initialisée mais pas configurée.
<code>NOT_INITIALIZED</code>	02 hex	La cartouche n'est pas initialisée.

Sous-chapitre 3.7

Types de données des fonctions système

Présentation

Cette section décrit les différents types de données des fonctions système de la bibliothèque PLCSystem de M241.

Contenu de ce sous-chapitre

Ce sous-chapitre contient les sujets suivants :

Sujet	Page
IMMEDIATE_ERR_TYPE : Codes d'erreur de la fonction <code>GetImmediateFastInput</code> de lecture des entrées d'E/S expertes intégrées	93
RTCSETDRIFT_ERROR : fonction <code>SetRTCDrift</code> - codes des erreurs détectées	94

IMMEDIATE_ERR_TYPE : Codes d'erreur de la fonction GetImmediateFastInput de lecture des entrées d'E/S expertes intégrées

Description du type énumération

Le type de données énumération contient les valeurs suivantes :

Enumérateur	Type	Commentaire
IMMEDIATE_NO_ERROR	WORD	Aucune erreur détectée.
IMMEDIATE_UNKNOWN	WORD	La référence de la fonction Immediate est incorrecte ou non configurée.
IMMEDIATE_UNKNOWN_PARAMETER	WORD	Une référence de paramètre est incorrecte.

RTCSETDRIFT_ERROR : fonction SetRTCDrift - codes des erreurs détectées**Description du type énumération**

Le type de données énumération RTCSETDRIFT_ERROR contient les valeurs suivantes :

Enumérateur	Valeur	Commentaire
RTC_OK	00 hex	L'écart de l'horodateur est configuré correctement.
RTC_BAD_DAY	01 hex	Non utilisé.
RTC_BAD_HOUR	02 hex	Non utilisé.
RTC_BAD_MINUTE	03 hex	Non utilisé.
RTC_BAD_DRIFT	04 hex	Paramètre d'écart de l'horodateur hors limites.
RTC_INTERNAL_ERROR	05 hex	Paramètres d'écart de l'horodateur rejetés sur détection d'une erreur interne.

Annexes



Annexe A

Représentation des fonctions et blocs fonction

Présentation

Chaque fonction peut être représentée dans les langages suivants :

- IL : (Instruction List) liste d'instructions
- ST : (Structured Text) littéral structuré
- LD : (Ladder Diagram) schéma à contacts
- FBD : Function Block Diagram (Langage à blocs fonction)
- CFC : Continuous Function Chart (Diagramme fonctionnel continu)

Ce chapitre fournit des exemples de représentations de fonctions et blocs fonction et explique comment les utiliser dans les langages IL et ST.

Contenu de ce chapitre

Ce chapitre contient les sujets suivants :

Sujet	Page
Différences entre une fonction et un bloc fonction	98
Utilisation d'une fonction ou d'un bloc fonction en langage IL	99
Utilisation d'une fonction ou d'un bloc fonction en langage ST	103

Différences entre une fonction et un bloc fonction

Fonction

Une fonction :

- est une POU (Program Organization Unit ou unité organisationnelle de programme) qui renvoie un résultat immédiat ;
- est directement appelée par son nom (et non par une instance) ;
- ne conserve pas son état entre deux appels ;
- peut être utilisée en tant qu'opérande dans des expressions.

Exemples : opérateurs booléens (AND), calculs, conversions (BYTE_TO_INT)

Bloc fonction

Un bloc fonction :

- est une POU qui renvoie une ou plusieurs sorties ;
- doit être appelé par une instance (copie de bloc fonction avec nom et variables dédiées).
- Chaque instance conserve son état (sorties et variables internes) entre deux appels à partir d'un bloc fonction ou d'un programme.

Exemples : temporisateurs, compteurs

Dans l'exemple, `Timer_ON` est une instance du bloc fonction `TON` :

```

1  PROGRAM MyProgram_ST
2  VAR
3      Timer_ON: TON; // Function Block Instance
4      Timer_RunCd: BOOL;
5      Timer_PresetValue: TIME := T#5S;
6      Timer_Output: BOOL;
7      Timer_ElapsedTime: TIME;
8  END_VAR

```

```

1  Timer_ON(
2      IN:=Timer_RunCd,
3      PT:=Timer_PresetValue,
4      Q=>Timer_Output,
5      ET=>Timer_ElapsedTime);

```

Utilisation d'une fonction ou d'un bloc fonction en langage IL

Informations générales

Cette partie explique comment mettre en œuvre une fonction et un bloc fonction en langage IL.

Les fonctions `IsFirstMastCycle` et `SetRTCDrift`, ainsi que le bloc fonction `TON`, sont utilisés à titre d'exemple pour illustrer les mises en œuvre.

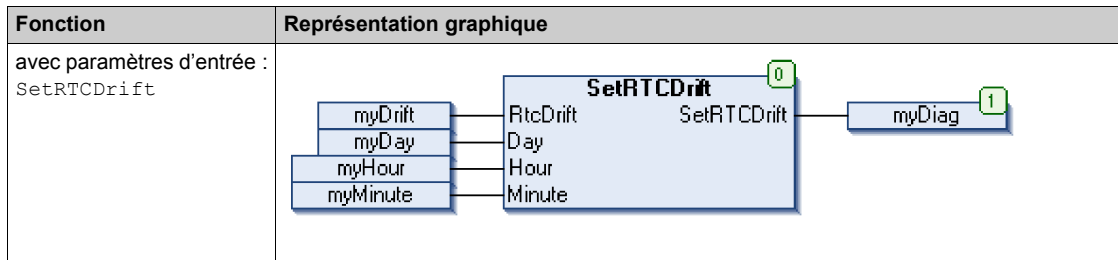
Utilisation d'une fonction en langage IL

La procédure suivante explique comment insérer une fonction en langage IL :

Etape	Action
1	Ouvrez ou créez un POU en langage IL (Instruction List, ou liste d'instructions). NOTE : La procédure de création d'un POU n'est pas détaillée ici. Pour plus d'informations, reportez-vous à la section Ajout et appel de POU (<i>voir SoMachine, Guide de programmation</i>).
2	Créez les variables nécessaires à la fonction.
3	Si la fonction possède une ou plusieurs entrées, chargez la première entrée en utilisant l'instruction LD.
4	Insérez une nouvelle ligne en dessous et : <ul style="list-style-type: none"> ● saisissez le nom de la fonction dans la colonne de l'opérateur (champ de gauche), ou ● utilisez l'Aide à la saisie pour choisir la fonction (sélectionnez Insérer l'appel de module dans le menu contextuel).
5	Si la fonction a plusieurs entrées et que l'Aide à la saisie est utilisée, le nombre requis de lignes est automatiquement créé avec ??? dans les champs situés à droite. Remplacez les ??? par la valeur ou la variable appropriée en fonction de l'ordre des entrées.
6	Insérez une nouvelle ligne pour stocker le résultat de la fonction dans la variable appropriée : saisissez l'instruction ST dans la colonne de l'opérateur (champ de gauche) et un nom de variable dans le champ situé à droite.

Pour illustrer la procédure, utilisons les fonctions `IsFirstMastCycle` (sans paramètre d'entrée) et `SetRTCDrift` (avec paramètres d'entrée) représentées graphiquement ci-après :

Fonction	Représentation graphique
sans paramètre d'entrée : <code>IsFirstMastCycle</code>	



En langage IL, le nom de la fonction est utilisé directement dans la colonne de l'opérateur :

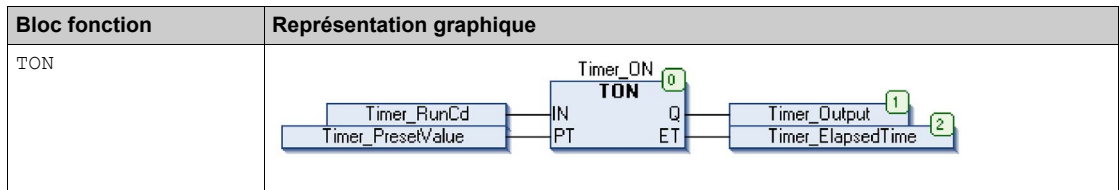
Fonction	Représentation dans l'éditeur IL de POU de SoMachine
Exemple IL d'une fonction sans paramètre d'entrée : IsFirstMastCycle	<pre> 1 PROGRAM MyProgram_IL 2 VAR 3 FirstCycle: BOOL; 4 END_VAR </pre> <hr/> <pre> 1 IsFirstMastCycle ST FirstCycle </pre>
Exemple IL d'une fonction avec des paramètres d'entrée : SetRTCDrift	<pre> 1 PROGRAM MyProgram_IL 2 VAR 3 myDrift: SINT (-29..29) := 5; 4 myDay: DAY_OF_WEEK := SUNDAY; 5 myHour: HOUR := 12; 6 myMinute: MINUTE; 7 myDiag: RTCSETDRIFT_ERROR; 8 END_VAR </pre> <hr/> <pre> 1 LD myDrift SetRTCDrift myDay myHour myMinute ST myDiag </pre>

Utilisation d'un bloc fonction en langage IL

La procédure suivante explique comment insérer un bloc fonction en langage IL :

Etape	Action
1	Ouvrez ou créez un POU en langage IL (Instruction List, ou liste d'instructions). NOTE : La procédure de création d'un POU n'est pas détaillée ici. Pour plus d'informations, reportez-vous à la section Ajout et appel de POU (<i>voir SoMachine, Guide de programmation</i>).
2	Créez les variables nécessaires au bloc fonction (y compris le nom de l'instance).
3	L'appel de blocs fonction nécessite l'utilisation d'une instruction <code>CAL</code> : <ul style="list-style-type: none"> • Utilisez l'Aide à la saisie pour sélectionner le bloc fonction (cliquez avec le bouton droit et sélectionnez Insérer l'appel de module dans le menu contextuel). • L'instruction <code>CAL</code> et les E/S nécessaires sont automatiquement créées. Chaque paramètre (E/S) est une instruction : <ul style="list-style-type: none"> • Les valeurs des entrées sont définies à l'aide de « := ». • Les valeurs des sorties sont définies à l'aide de « => ».
4	Dans le champ <code>CAL</code> de droite, remplacez les ??? par le nom de l'instance.
5	Remplacez les autres ??? par une variable ou une valeur immédiate appropriée.

Pour illustrer la procédure, utilisons le bloc fonction `TON` représenté graphiquement ci-après :



En langage IL, le nom du bloc fonction est utilisé directement dans la colonne de l'opérateur :

Bloc fonction	Représentation dans l'éditeur IL de POU de SoMachine
TON	<pre>1 PROGRAM MyProgram_IL 2 VAR 3 Timer_ON: TON; // Function Block instance declaration 4 Timer_RunCd: BOOL; 5 Timer_PresetValue: TIME := T#5S; 6 Timer_Output: BOOL; 7 Timer_ElapsedTime: TIME; 8 END_VAR 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230 231 232 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255 256 257 258 259 260 261 262 263 264 265 266 267 268 269 270 271 272 273 274 275 276 277 278 279 280 281 282 283 284 285 286 287 288 289 290 291 292 293 294 295 296 297 298 299 300 301 302 303 304 305 306 307 308 309 310 311 312 313 314 315 316 317 318 319 320 321 322 323 324 325 326 327 328 329 330 331 332 333 334 335 336 337 338 339 340 341 342 343 344 345 346 347 348 349 350 351 352 353 354 355 356 357 358 359 360 361 362 363 364 365 366 367 368 369 370 371 372 373 374 375 376 377 378 379 380 381 382 383 384 385 386 387 388 389 390 391 392 393 394 395 396 397 398 399 400 401 402 403 404 405 406 407 408 409 410 411 412 413 414 415 416 417 418 419 420 421 422 423 424 425 426 427 428 429 430 431 432 433 434 435 436 437 438 439 440 441 442 443 444 445 446 447 448 449 450 451 452 453 454 455 456 457 458 459 460 461 462 463 464 465 466 467 468 469 470 471 472 473 474 475 476 477 478 479 480 481 482 483 484 485 486 487 488 489 490 491 492 493 494 495 496 497 498 499 500 501 502 503 504 505 506 507 508 509 510 511 512 513 514 515 516 517 518 519 520 521 522 523 524 525 526 527 528 529 530 531 532 533 534 535 536 537 538 539 540 541 542 543 544 545 546 547 548 549 550 551 552 553 554 555 556 557 558 559 560 561 562 563 564 565 566 567 568 569 570 571 572 573 574 575 576 577 578 579 580 581 582 583 584 585 586 587 588 589 590 591 592 593 594 595 596 597 598 599 600 601 602 603 604 605 606 607 608 609 610 611 612 613 614 615 616 617 618 619 620 621 622 623 624 625 626 627 628 629 630 631 632 633 634 635 636 637 638 639 640 641 642 643 644 645 646 647 648 649 650 651 652 653 654 655 656 657 658 659 660 661 662 663 664 665 666 667 668 669 670 671 672 673 674 675 676 677 678 679 680 681 682 683 684 685 686 687 688 689 690 691 692 693 694 695 696 697 698 699 700 701 702 703 704 705 706 707 708 709 710 711 712 713 714 715 716 717 718 719 720 721 722 723 724 725 726 727 728 729 730 731 732 733 734 735 736 737 738 739 740 741 742 743 744 745 746 747 748 749 750 751 752 753 754 755 756 757 758 759 760 761 762 763 764 765 766 767 768 769 770 771 772 773 774 775 776 777 778 779 780 781 782 783 784 785 786 787 788 789 790 791 792 793 794 795 796 797 798 799 800 801 802 803 804 805 806 807 808 809 810 811 812 813 814 815 816 817 818 819 820 821 822 823 824 825 826 827 828 829 830 831 832 833 834 835 836 837 838 839 840 841 842 843 844 845 846 847 848 849 850 851 852 853 854 855 856 857 858 859 860 861 862 863 864 865 866 867 868 869 870 871 872 873 874 875 876 877 878 879 880 881 882 883 884 885 886 887 888 889 890 891 892 893 894 895 896 897 898 899 900 901 902 903 904 905 906 907 908 909 910 911 912 913 914 915 916 917 918 919 920 921 922 923 924 925 926 927 928 929 930 931 932 933 934 935 936 937 938 939 940 941 942 943 944 945 946 947 948 949 950 951 952 953 954 955 956 957 958 959 960 961 962 963 964 965 966 967 968 969 970 971 972 973 974 975 976 977 978 979 980 981 982 983 984 985 986 987 988 989 990 991 992 993 994 995 996 997 998 999 1000</pre>

Utilisation d'une fonction ou d'un bloc fonction en langage ST

Informations générales

Cette partie décrit comment mettre en œuvre une fonction ou un bloc fonction en langage ST.

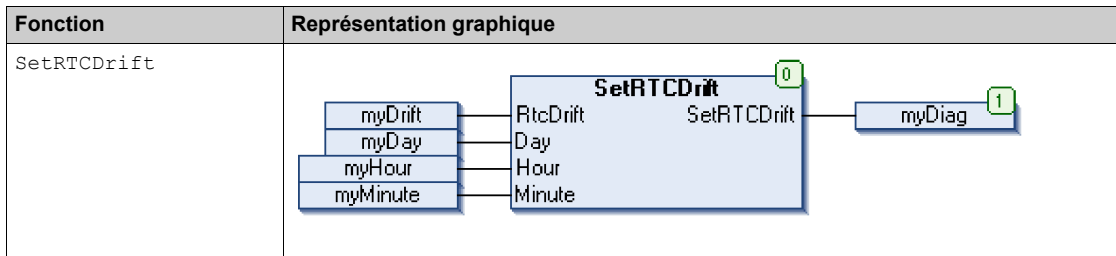
La fonction `SetRTCDrift` et le bloc fonction `TON` sont utilisés à titre d'exemple pour illustrer les mises en œuvre.

Utilisation d'une fonction en langage ST

La procédure suivante explique comment insérer une fonction en langage ST :

Etape	Action
1	Ouvrez ou créez un POU en langage ST (Structured Text ou Littéral structuré). NOTE : La procédure de création d'un POU n'est pas détaillée ici. Pour plus d'informations, reportez-vous à la section Ajout et appel de POU (<i>voir SoMachine, Guide de programmation</i>).
2	Créez les variables nécessaires à la fonction.
3	Utilisez la syntaxe générale dans l' éditeur ST de POU pour la représentation en langage ST d'une fonction. La syntaxe générale est la suivante : RésultatFonction:= NomFonction(VarEntrée1, VarEntrée2, ... VarEntréex);

Pour illustrer la procédure, utilisons la fonction `SetRTCDrift` représentée graphiquement ci-après :



La représentation en langage ST de cette fonction est la suivante :


Fonction	Représentation dans l'éditeur ST de POU de SoMachine
SetRTCDrift	<pre>PROGRAM MyProgram_ST VAR myDrift: SINT(-29..29) := 5; myDay: DAY_OF_WEEK := SUNDAY; myHour: HOUR := 12; myMinute: MINUTE; myRTCAdjust: RTCDRIFT_ERROR; END_VAR myRTCAdjust:= SetRTCDrift(myDrift, myDay, myHour, myMinute);</pre>

Utilisation d'un bloc fonction en langage ST

La procédure suivante explique comment insérer un bloc fonction en langage ST :

Etap e	Action
1	Ouvrez ou créez un POU en langage IL (Instruction List, ou liste d'instructions). NOTE : La procédure de création d'un POU n'est pas détaillée ici. Pour plus d'informations sur l'ajout, la déclaration et l'appel de POU, reportez-vous à la documentation (<i>voir SoMachine, Guide de programmation</i>) associée.
2	Créez les variables d'entrée, les variables de sortie et l'instance requises pour le bloc fonction : <ul style="list-style-type: none"> ● Les variables d'entrée sont les paramètres d'entrée requis par le bloc fonction. ● Les variables de sortie reçoivent la valeur renvoyée par le bloc fonction.
3	Utilisez la syntaxe générale dans l' éditeur ST de POU pour la représentation en langage ST d'un bloc fonction. La syntaxe générale est la suivante : BlocFonction_NomInstance (Entrée1:=VarEntrée1, Entrée2:=VarEntrée2,... Sortie1=>VarSortie1, Sortie2=>VarSortie2,...);

Pour illustrer la procédure, utilisons le bloc fonction TON représenté graphiquement ci-après :

Bloc fonction	Représentation graphique
TON	

Le tableau suivant montre plusieurs exemples d'appel de bloc fonction en langage ST :

Bloc fonction	Représentation dans l'éditeur ST de POU de SoMachine
TON	<pre>1 PROGRAM MyProgram_ST 2 VAR 3 Timer_ON: TON; // Function Block Instance 4 Timer_RunCd: BOOL; 5 Timer_PresetValue: TIME := T#5S; 6 Timer_Output: BOOL; 7 Timer_ElapsedTime: TIME; 8 END_VAR 1 Timer_ON(2 IN:=Timer_RunCd, 3 PT:=Timer_PresetValue, 4 Q=>Timer_Output, 5 ET=>Timer_ElapsedTime);</pre>



0-9

%MW

Selon la norme IEC, %MW représente un registre de mots mémoire (par exemple, un objet langage de type mot mémoire).

(%PLS)

Unité de programmation qui possède 1 ou plusieurs entrées et renvoie 1 ou plusieurs sorties. Les blocs fonction (FBs) sont appelés via une instance (copie du bloc fonction avec un nom et des variables dédiés) et chaque instance a un état persistant (sorties et variables internes) d'un appel au suivant.

Exemples : temporisateurs, compteurs

A

adresse MAC

(*media access control address*) Numéro unique sur 48 bits associé à un élément matériel spécifique. L'adresse MAC est programmée dans chaque carte réseau ou équipement lors de la fabrication.

application

Programme comprenant des données de configuration, des symboles et de la documentation.

application de démarrage

(*boot application*). Fichier binaire qui contient l'application. En général, il est stocké dans le contrôleur (PLC) et permet au PLC de démarrer sur l'application que l'utilisateur a générée.

ARRAY

Agencement systématique d'objets de données d'un même type sous la forme d'un tableau défini dans la mémoire d'un Logic Controller. La syntaxe est la suivante : `ARRAY [<dimension>] OF <Type>`

Exemple 1 : `ARRAY [1..2] OF BOOL` est un tableau à 1 dimension composé de 2 éléments de type `BOOL`.

Exemple 2 : `ARRAY [1..10, 1..20] OF INT` est un tableau à 2 dimensions composés de 10 x 20 éléments de type `INT`.

B**BOOL**

(*booléen*) Type de données informatique standard. Une variable de type `BOOL` peut avoir l'une des deux valeurs suivantes : 0 (`FALSE`), 1 (`TRUE`). Un bit extrait d'un mot est de type `BOOL` ; par exemple, `%MW10.4` est le cinquième bit d'un mot mémoire numéro 10.

BOOTP

Acronyme de *bootstrap protocol*. Protocole réseau UDP qu'un client réseau peut utiliser pour obtenir automatiquement une adresse IP (et éventuellement d'autres données) à partir d'un serveur. Le client s'identifie auprès du serveur à l'aide de son adresse MAC. Le serveur, qui gère un tableau préconfiguré des adresses MAC des équipements client et des adresses IP associées, envoie au client son adresse IP préconfigurée. A l'origine, le protocole BOOTP était utilisé pour amorcer à distance les hôtes sans lecteur de disque à partir d'un réseau. Le processus BOOTP affecte une adresse IP de durée illimitée. Le service BOOTP utilise les ports UDP 67 et 68.

C**CFC**

Acronyme de *continuous function chart*, diagramme fonctionnel continu. Langage de programmation graphique (extension de la norme IEC 61131-3) basé sur le langage de diagramme à blocs fonction et qui fonctionne comme un diagramme de flux. Toutefois, il n'utilise pas de réseaux et le positionnement libre des éléments graphiques est possible, ce qui permet les boucles de retour. Pour chaque bloc, les entrées se situent à gauche et les sorties à droite. Vous pouvez lier les sorties de blocs aux entrées d'autres blocs pour créer des expressions complexes.

chaîne

Variable composée d'une série de caractères ASCII.

chien de garde

Temporisateur spécial utilisé pour garantir que les programmes ne dépassent pas le temps de scrutation qui leur est alloué. Le chien de garde est généralement réglé sur une valeur supérieure au temps de scrutation et il est remis à 0 à la fin de chaque cycle de scrutation. S'il atteint la valeur prédéfinie (par exemple, parce que le programme est bloqué dans une boucle sans fin) une erreur est déclarée et le programme s'arrête.

configuration

Agencement et interconnexions des composants matériels au sein d'un système, ainsi que les paramètres matériels et logiciels qui déterminent les caractéristiques de fonctionnement du système.

contrôleur

Automatise des processus industriels. On parle également de contrôleur logique programmable (PLC) ou de contrôleur programmable.

CRC

Contrôle de redondance cyclique. Méthode utilisée pour déterminer la validité d'une transmission de communication. La transmission contient un champ de bits qui constitue un total de contrôle. Le message est utilisé pour le calcul de ce total de contrôle par l'émetteur en fonction du contenu du message. Les noeuds récepteurs recalculent ensuite ce champ de la même manière. Tout écart entre les deux calculs de CRC indique que le message émis et le message reçu sont différents.

D**DHCP**

(dynamic host configuration protocol). Extension avancée du protocole BOOTP. Bien que DHCP soit plus avancé, DHCP et BOOTP sont tous les deux courants. (Le protocole DHCP peut gérer des demandes de client BOOTP.)

DWORD

Abréviation de *double word*, mot double. Codé au format 32 bits.

E**E/S**

Entrée/sortie

élément

Raccourci pour l'élément d'un ARRAY.

équipement

Partie d'une machine comprenant des sous-ensembles tels que des transporteurs, des plaques tournantes, etc.

Ethernet

Technologie de couche physique et de liaison de données pour les réseaux locaux (LANs) également appelée IEE 802.3.

EtherNet/IP

Acronyme de *Ethernet Industrial Protocol*, protocole industriel Ethernet. Protocole de communication ouvert pour les solutions d'automatisation de la production dans les systèmes industriels. EtherNet/IP est une famille de réseaux mettant en œuvre le protocole CIP au niveau des couches supérieures. L'organisation ODVA spécifie qu'EtherNet/IP permet une adaptabilité générale et une indépendance des supports.

F**FB**

Acronyme de *function block*, bloc fonction. Mécanisme de programmation commode qui consolide un groupe d'instructions de programmation visant à effectuer une action spécifique et normalisée telle que le contrôle de vitesse, le contrôle d'intervalle ou le comptage. Un bloc fonction peut comprendre des données de configuration, un ensemble de paramètres de fonctionnement interne ou externe et généralement une ou plusieurs entrées et sorties de données.

fonction

Unité de programmation possédant 1 entrée et renvoyant 1 résultat immédiat. Contrairement aux blocs fonction (FBs), une fonction est appelée directement par son nom (et non via une instance), elle n'a pas d'état persistant d'un appel au suivant et elle peut être utilisée comme opérande dans d'autres expressions de programmation.

Exemples : opérateurs booléens (AND), calculs, conversion (BYTE_TO_INT).

G**GVL**

Acronyme de *global variable list*, liste de variables globales. Gère les variables globales qui peuvent être transmises entre contrôleurs sur un réseau Ethernet TCP/IP Modbus.

H**hex**

(*hexadécimal*)

I**ID**

(*identificateur/identification*)

IEC

Acronyme *International Electrotechnical Commission*, Commission Electrotechnique Internationale (CEI). Organisation internationale non gouvernementale à but non lucratif, qui rédige et publie les normes internationales en matière d'électricité, d'électronique et de domaines connexes.

IEC 61131-3

Partie 3 d'une norme en 3 parties de l'IEC pour les équipements d'automatisation industriels. La norme IEC 61131-3 traite des langages de programmation des contrôleurs. Elle définit 2 normes pour la programmation graphique et 2 normes pour la programmation textuelle. Les langages de programmation graphiques sont le schéma à contacts (LD) et le langage à blocs fonction (FBD). Les langages textuels comprennent le texte structuré (ST) et la liste d'instructions (IL).

IEEE 802.3

Ensemble de normes IEEE définissant la couche physique et la sous-couche de la couche de liaison de données de l'Ethernet câblé.

IL

Acronyme de *instruction list*, liste d'instructions. Un programme écrit en langage IL est composé d'instructions textuelles qui sont exécutées séquentiellement par le contrôleur. Chaque instruction comprend un numéro de ligne, un code d'instruction et un opérande (voir la norme IEC 61131-3).

INT

Abréviation de *integer*), nombre entier codé sur 16 bits.

IP

Acronyme de *Internet Protocol*, protocole Internet. Le protocole IP fait partie de la famille de protocoles TCP/IP, qui assure le suivi des adresses Internet des équipements, achemine les messages sortants et reconnaît les messages entrants.

L**Langage en blocs fonctionnels**

Un des 5 langages de programmation de logique ou de commande pris en charge par la norme IEC 61131-3 pour les systèmes de commande. FBD est un langage de programmation orienté graphique. Il fonctionne avec une liste de réseaux où chaque réseau contient une structure graphique de zones et de lignes de connexion représentant une expression logique ou arithmétique, un appel de bloc fonction ou une instruction de retour.

LD

Acronyme de *ladder diagram*, schéma à contacts. Représentation graphique des instructions d'un programme de contrôleur, avec des symboles pour les contacts, les bobines et les blocs dans une série de réseaux exécutés séquentiellement par un contrôleur (voir IEC 61131-3).

LWORD

Acronyme de *long word*, mot long. Type de données codé sur 64 bits.

M**MAST**

Tâche de processeur exécutée par le biais de son logiciel de programmation. La tâche MAST comprend deux parties :

- **IN** : les entrées sont copiées dans la section IN avant exécution de la tâche MAST.
- **OUT** : les sorties sont copiées dans la section OUT après exécution de la tâche MAST.

mémoire flash

Mémoire non volatile qui peut être écrasée. Elle est stockée dans une puce EEPROM spéciale, effaçable et reprogrammable.

micrologiciel

Représente le BIOS, les paramètres de données et les instructions de programmation qui constituent le système d'exploitation d'un contrôleur. Le micrologiciel est stocké dans la mémoire non volatile du contrôleur.

O

octet

Type codé sur 8 bits, de 16#00 à 16#FF en représentation hexadécimale.

P

PLC

Acronyme de *programmable logic controller*, contrôleur logique programmable. Ordinateur industriel utilisé pour automatiser des processus de fabrication et autres processus électromécaniques. Les contrôleurs PLCs diffèrent des ordinateurs courants par le fait qu'ils sont conçus pour utiliser plusieurs tableaux d'entrées et de sorties et pour accepter des conditions de choc, de vibration, de température et d'interférences électriques plus rudes.

POU

Acronyme de *program organization unit*, unité organisationnelle de programme. Déclaration de variables dans le code source et jeu d'instructions correspondant. Les POU facilitent la réutilisation modulaire de programmes logiciels, de fonctions et de blocs fonction. Une fois déclarées, les POU sont réutilisables.

programme

Composant d'une application constitué de code source compilé qu'il est possible d'installer dans la mémoire d'un Logic Controller.

protocole

Convention ou définition standard qui contrôle ou permet la connexion, la communication et le transfert de données entre 2 systèmes informatiques et leurs équipements.

R

réseau

Système d'équipements interconnectés qui partagent un chemin de données et un protocole de communications communs.

run

Commande qui ordonne au contrôleur de scruter le programme d'application, lire les entrées physiques et écrire dans les sorties physiques en fonction de la solution de la logique du programme.

S

ST

Acronyme de *structured text*, texte structuré. Langage composé d'instructions complexes et d'instructions imbriquées (boucles d'itération, exécutions conditionnelles, fonctions). Le langage ST est conforme à la norme IEC 61131-3.

STOP

Commande ordonnant au contrôleur de cesser d'exécuter un programme d'application.

T

tâche

Ensemble de sections et de sous-programmes, exécutés de façon cyclique ou périodique pour la tâche MAST, ou périodique pour la tâche FAST.

Une tâche présente un niveau de priorité et des entrées et sorties du contrôleur lui sont associées. Ces E/S sont actualisées par rapport à la tâche.

Un contrôleur peut comporter plusieurs tâches.

TCP

Acronyme de *transmission control protocol*, protocole de contrôle de transmission. Protocole de couche de transport basé sur la connexion qui assure la transmission de données simultanée dans les deux sens. TCP fait partie de la suite de protocoles TCP/IP.

U

UDINT

Abréviation de *unsigned double integer*, entier double non signé. Valeur codée sur 32 bits.

UINT

Abréviation de *unsigned integer*, entier non signé. Valeur codée sur 16 bits.

V

variable

Unité de mémoire qui est adressée et modifiée par un programme.

variable non localisée

Variable qui n'a pas d'adresse (voir *variable localisée*).

variable système

Variable qui fournit des données de contrôleur et des informations de diagnostic et permet d'envoyer des commandes au contrôleur.

W

WORD

Type de données codé sur 16 bits.



Specials

C

- CART_R_ARRAY_TYPE
 - types de données, 89
- CART_R_MODULE_ID
 - types de données, 90
- CART_R_STATE
 - types de données, 91
- CART_R_STRUCT
 - variables système, 38

D

- DataFileCopy
 - fonctions, 52
- DataFileCopyError
 - types de données, 73
- DataFileCopyLocation
 - types de données, 74

E

- ETH_R
 - variable système, 30
- ETH_R_FRAME_PROTOCOL
 - types de données, 77
- ETH_R_IP_MODE
 - types de données, 78
- ETH_R_PORT_DUPLEX_STATUS
 - types de données, 79
- ETH_R_PORT_LINK_STATUS
 - types de données, 81
- ETH_R_PORT_SPEED
 - types de données, 82
- ETH_W
 - variable système, 35
- ExecuteScript
 - fonctions, 55

- ExecuteScriptError
 - types de données, 75

F

- fonctions
 - DataFileCopy, 52
 - différences entre une fonction et un bloc fonction, 98
 - ExecuteScript, 55
 - GetImmediateFastInput, 41
 - GetRtc, 42
 - IsFirstMastColdCycle, 43
 - IsFirstMastCycle, 44
 - IsFirstMastWarmCycle, 46
 - PhysicalWriteFastOutputs, 48
 - SetRTCDrift, 49
 - TM3_GetModuleBusStatus, 58
 - TM3_GetModuleInternalStatus, 59
 - utilisation d'une fonction ou d'un bloc fonction en langage IL, 99
 - utilisation d'une fonction ou d'un bloc fonction en langage ST, 103

G

- GetImmediateFastInput
 - fonctions, 41
- GetRtc
 - fonctions, 42

I

- IMMEDIATE_ERR_TYPE
 - types de données, 93
- IsFirstMastColdCycle
 - fonctions, 43
- IsFirstMastCycle
 - fonctions, 44
- IsFirstMastWarmCycle
 - fonctions, 46

P

PhysicalWriteFastOutputs
fonctions, 48

PLC_R
variable système, 20

PLC_R_APPLICATION_ERROR
types de données, 63

PLC_R_BOOT_PROJECT_STATUS
types de données, 64

PLC_R_SDCARD_STATUS
types de données, 66

PLC_R_STATUS
types de données, 67

PLC_R_STOP_CAUSE
types de données, 68

PLC_R_TERMINAL_PORT_STATUS
types de données, 69

PLC_R_TM3_BUS_STATE
types de données, 70

PLC_W
variable système, 25

PLC_W_COMMAND
types de données, 71

PROFIBUS_R
System Variable, 37

R

RTCSETDRIFT_ERROR
types de données, 94

S

SERIAL_R
variable système, 27

SERIAL_W
variable système, 28

SetRTCDrift
fonctions, 49

System variable
PROFIBUS_R, 37

T

TM3_ERR_CODE
types de données, 85

TM3_GetModuleBusStatus
fonctions, 58

TM3_GetModuleInternalStatus
fonctions, 59

TM3_MODULE_R
variables système, 36

TM3_MODULE_R_ARRAY_TYPE
types de données, 86

TM3_MODULE_STATE
types de données, 87

types de données

- CART_R_ARRAY_TYPE, 89
- CART_R_MODULE_ID, 90
- CART_R_STATE, 91
- DataFileCopyError, 73
- DataFileCopyLocation, 74
- ETH_R_FRAME_PROTOCOL, 77
- ETH_R_IP_MODE, 78
- ETH_R_PORT_DUPLEX_STATUS, 79
- ETH_R_PORT_IP_STATUS, 80
- ETH_R_PORT_LINK_STATUS, 81
- ETH_R_PORT_SPEED, 82
- ETH_R_RUN_IDLE, 83
- ExecuteScriptError, 75
- IMMEDIATE_ERR_TYPE, 93
- PLC_R_APPLICATION_ERROR, 63
- PLC_R_BOOT_PROJECT_STATUS, 64
- PLC_R_IO_STATUS, 65
- PLC_R_SDCARD_STATUS, 66
- PLC_R_STATUS, 67
- PLC_R_STOP_CAUSE, 68
- PLC_R_TERMINAL_PORT_STATUS, 69
- PLC_R_TM3_BUS_STATE, 70
- PLC_W_COMMAND, 71
- RTCSETDRIFT_ERROR, 94
- TM3_ERR_CODE, 85
- TM3_MODULE_R_ARRAY_TYPE, 86
- TM3_MODULE_STATE, 87

V

variable système

ETH_R, 30

ETH_W, 35

PLC_R, 20

PLC_W, 25

SERIAL_R, 27

SERIAL_W, 28

variables système

CART_R_STRUCT, 38

définition, 15

TM3_MODULE_R, 36

Variables système

Utilisation, 17

