RightWON protocole CANbus/CANopen - Manuel - V1.4

Document No. RWM002015-MA-fr

vizimax

www.vizimax.com

Copyright

Intégrité de l'information

Vizimax considère que les informations contenues dans ce document sont exactes au moment de sa publication. Toutefois, ce document peut contenir des erreurs ou omissions. Vizimax n'offre aucune garantie concernant le présent document ou son contenu. En aucun cas, Vizimax ne peut être tenu pour responsable de quelconques pertes ou dommages de toute nature découlant de l'utilisation de ce document ou des informations qu'il contient. De plus, ce document ou les informations qu'il contient ne peuvent être considérées comme opposables à Vizimax ou utilisées ou retenues contre Vizimax. L'information peut être périodiquement mise à jour ou modifiée sans préavis dans les éditions ultérieures du document. Si vous découvrez une erreur dans ce document, nous vous prions de la signaler à Vizimax.

Toutes représentations ou déclarations contenues dans ce document concernant les produits Vizimax sont à des fins informatives seulement et ne constituent pas une garantie, expresse ou implicite, concernant ces produits. La garantie limitée standard de Vizimax, formulée dans le contrat de vente ou de la confirmation de commande, est la seule garantie offerte par Vizimax et applicable aux produits.

Toutes les spécifications et conceptions sont sujettes à des changements sans préavis. Vizimax se réserve le droit, à sa seule discrétion, de modifier ou de remplacer une partie du présent document. Il est de votre responsabilité de vérifier périodiquement si des mises à jour du document sont disponibles.

Exclusion de garantie

Vizimax, ses fournisseurs et concédants excluent par les présentes toute garantie, expresse ou implicite, y compris, sans s'y limiter, les garanties d'adéquation à un usage particulier et de non-contrefaçon. Ni Vizimax ni ses fournisseurs et concédants de licence, n'offrent la garantie que les produits Vizimax seront sans erreur, ni que l'accès à des unités déportées à distance et des équipements qui y sont connectés sera continu ou ininterrompu.

Limite de responsabilité

Les produits Vizimax sont des dispositifs programmables et paramétrables qui peuvent être modifiés par tout utilisateur ayant accès à un logiciel de configuration et qui a reçu l'autorisation d'accéder au produit. Vizimax ne peut pas surveiller les modifications à la configuration de produits Vizimax à moins qu'une entente de service préalable ait été conclue entre toutes les parties concernées. Vizimax n'a aucun contrôle sur les droits d'accès à des produits Vizimax. Vizimax ne saurait donc être tenu pour responsable de la configuration, des automatismes et des actions qui sont programmées dans tout produit Vizimax une fois qu'il a été livré à l'acheteur ou un tiers. De même, Vizimax n'est pas responsable de l'usage particulier des produits Vizimax dans les applications industrielles, commerciales ou autres, et il n'est pas responsable des éventuels effets nocifs découlant de cette utilisation.

Vous avez la responsabilité de prendre les précautions nécessaires pour vous protéger, protéger vos réseaux informatiques et tous les équipements qui y sont connectés contre toute action nuisible ou destructrice qui découle d'une programmation incorrecte des produits Vizimax ou découlant d'une action volontaire ou involontaire d'un utilisateur. Vizimax décline toute responsabilité pour tout préjudice résultant de l'utilisation de la RightWON.

En aucun cas, Vizimax, ses fournisseurs ou concédants de licence, ne peuvent être tenus responsable à l'égard de toute question soumise en vertu de tout contrat, de négligence, responsabilité stricte ou autre théorie juridique ou équitable pour: (i) des dommages spéciaux, directs ou indirects, (ii) le coût des achats ou des produits de substitution ou services, (iii) l'interruption de service ou la perte ou corruption de données.

Vizimax, ses entrepreneurs, donneurs de licence, ainsi que leurs administrateurs, dirigeants, employés et agents, sont indemnisés de toute réclamation et frais, y compris les honoraires d'avocats, résultant de votre utilisation ou mauvaise utilisation des produits Vizimax. Vizimax n'assume aucune responsabilité pour tout dommage, blessure, défaut de fonctionnement ou retard dû à des questions au-delà du contrôle raisonnable de Vizimax.

Ce qui précède ne s'applique pas dans la mesure où la loi applicable l'interdit.

Marques de commerce

Vizimax, le logo Vizimax, RightWON, WiseWON, SynchroTeq, SynchroTeq+ et les icônes RightWON sont des marques de commerce ou des marques déposées de Vizimax Inc. au Canada, aux États-Unis ainsi que d'autres juridictions. Toutes les autres marques de commerce, marques déposées et marques de service sont la propriété de leurs propriétaires respectifs.

Votre utilisation des produits Vizimax ne vous donne aucun droit ou licence de reproduction ou d'utilisation des marques de commerce Vizimax ou de ses tierce parties.

Vizimax est un usager licencié des marques de commerce suivantes:

COPALP STRATON atvise®

Table des matières

Historique des révisions

Date (aa-mm-jj)	**Commentaires**	**Auteur**
2011-08-08	V1.0 : Relâche préliminaire	P. Taillefer
2011-08-09	V1.1 : Ajout d'informations sur les coupleurs CANopen	P. Taillefer
2011-11-15	V1.2 : Mise à jour des documents applicables	C. Archambault
2012-01-11	V1.3 : Déplacer les sections sur le paramétrage du pilote et du port CANbus de la section Intégration du protocole CANbus à la section Tutoriel de configuration du CANbus.	C. Archambault
2012-02-10	V1.4 : Ajout d'informations dans les sections *Séquence de définition et d'assignation d'un PDO* et *Recette de commandes.*	C. Archambault

Applicabilité de la documentation

La présente documentation est applicable aux versions logicielles suivantes du RightWON Configuration Suite :

Version de la documentation	**Version du produit**	**Commentaires**
V1.0 et plus	1.7 et plus	S'applique à la version 1.7 et plus du logiciel RightWON ⚠

Note importante ⚠

Il est recommandé aux utilisateurs de systèmes CANopen de passer à la version 1.7 ou plus récente du logiciel embarqué du RightWON car les problèmes suivants ont été détectés dans les versions antérieures :

- Lorsque plusieurs requêtes de lecture des données analogiques TPD02 à TPD04 sont lancées simultanément, les variables ne sont pas toujours rafraîchies correctement. Les requêtes de lecture périodique ou sur demande doivent être décalées;
- Les requêtes d'écriture comportant 4 octets (SDO) ne sont pas exécutées correctement. Les commandes touchant l'initialisation des modules, l'assignation et le mapping des PDO pourraient ne pas fonctionner correctement.

Introduction

Nous vous remercions pour la confiance que vous nous témoignez, et vous félicitons d'avoir choisi le système RightWON de Vizimax ! Pour votre satisfaction et la réussite de vos projets, le système RightWON a été conçu et fabriqué selon les plus hauts standards de qualité et de performance de l'industrie.

1.1. Portée du document

Ce document décrit l'intégration du protocole de communication CANbus/CANopen dans un système RightWON à l'aide du logiciel RightWON Configuration Suite.

1.1.1. Documents applicables

Pour approfondir les informations fournies dans ce document, le lecteur pourra se reporter aux manuels spécialisés suivants :

Références	Liste de documents
RWM000010-MA	RightWON Configuration Suite - Manuel
RWM000011-MA	RightWON Configuration Suite - Guide d'installation
RWM000020-MA	RightWON - Guide d'opération
RWM000050-MA	RightWON Satellite – Manuel de l'utilisateur
RWM000080-MA	RightWON Configuration Suite - Guide d'application

1.2. Conventions du document

Pour faciliter la lecture du document, les conventions suivantes sont utilisées :

- les commandes et items de menus/dialogues (**Options/paramètres avancé…)** et boutons **OK** sont en **caractères gras;**
- les noms définis par les intégrateurs sont en *caractères italiques;*
- les entités spécifiques des applications telles que Secteur, Tag, Catégorie débutent par une lettre Majuscule;
- les hyperliens sont en couleur bleue;
- le symbole ⚠ est utilisé pour attirer l'attention du lecteur.

1.3. Directives de sécurité

Dans le but d'assurer la sécurité des personnes et des biens, et de prévenir les risques d'accident, observez attentivement les avertissements et mises en garde inscrites sur les produits, manuels et emballages des produits RightWON.

Dans le but d'assurer un fonctionnement correct du produit RightWON, lisez ce manuel en totalité avant de procéder aux autres étapes d'apprentissage, d'installation du matériel, de configuration ou d'opération. Assurez-vous de comprendre le produit et les informations contenues dans ce manuel. Pour aller plus loin ou pour bénéficier d'une assistance de Vizimax, écrivez à nos ingénieurs d'applications ou à notre support technique à l'adresse support@vizimax.com (certains frais et conditions peuvent s'appliquer en fonction de la nature des services attendus).

1.3.1. Avertissements ⚠

Les produits RightWON ne sont pas conçus pour des applications de gestion de la sécurité ou comme dispositifs de sécurité. Une utilisation inadéquate des produits peut engendrer des situations critiques entraînant des dommages aux personnes, aux biens et équipements, des défauts de réseaux informatiques, des pertes de données, des chocs électriques, des blessures sérieuses et même la mort. Afin de prévenir l'apparition de tels évènements :

- Prendre toutes les mesures nécessaires pour assurer la sécurité de vos systèmes en utilisant des équipements appropriés rencontrant les caractéristiques requises par l'application. Ceci vous aidera à préserver l'intégrité de vos systèmes en cas de défaut de fonctionnement ou d'autres facteurs externes.
- Pour prévenir les risques d'explosion, ne pas utiliser les produits RightWON dans les environnements explosifs sans prendre les mesures appropriées définies par les normes et règlements en vigueur obtenus auprès des autorités locales compétentes.
- Pour prévenir les dommages au matériel électronique, ne pas exposer le produit à une flamme directe et ne soumettez pas le produit à des contraintes environnementales dépassant les spécifications.
- Les piles peuvent exploser si elles ne sont pas manipulées avec soin. Ne pas les recharger, désassembler ou les jeter au feu. Nous vous recommandons de recycler ces éléments en les mettant à la disposition des réseaux de collecte appropriés.

1.3.2. Mises en garde ⚠

- Assurez vous que les produits RightWON sont gérés par du personnel qualifié qui a été adéquatement formé pour l'installer, le configurer ou le dépanner.
- Veillez à toujours configurer et exploiter les produits de façon à ne pas excéder les caractéristiques techniques et critères d'opération recommandés par Vizimax, énoncés dans ce manuel et dans les autres documents de spécification disponibles.
- Utilisez des dispositifs d'urgence externes et homologués incluant sans s'y limiter : arrêts d'urgence, signalisations d'urgence, circuits de verrouillage et de sécurité.
- Attachez et verrouillez les câbles et connecteurs débrochables. Des connecteurs mal raccordés peuvent générer de la surchauffe et prendre feu.
- Protégez toutes les alimentations électriques et branchez un conducteur de mise à la terre sur l'équipement en utilisant une connexion appropriée. Un défaut de protection et/ou de mise à la terre de l'équipement peut causer des chocs électriques mortels.
- Prenez toutes les précautions utiles pour empêcher les matières étrangères de pénétrer à l'intérieur du produit (liquides, matériel inflammable, objets métalliques, etc.).
- Mettez l'équipement hors tension et déconnectez toutes les sources d'alimentation avant d'entreprendre quelques travaux que ce soit sur l'équipement.

Gestion du protocole CANbus- CANopen

Le CANbus est un bus utilisé pour échanger des données de façon fiable entre des partenaires sur un lien de communication série. Les sujets suivants sont traités dans ce document :

1- Introduction au protocole CANbus, définissant un nœud, un message CAN, l'arbitrage du bus, les niveaux de mise en œuvre et intégration du protocole CANopen dans le CANbus.

2- Intégration du protocole CANbus dans le RightWON, décrivant les interrelations entre l'applicatif PLC IEC 61131-3, le gestionnaire de bus de terrain, le lien du gestionnaire réseau et le matériel.

3- Tutoriel de configuration du CANbus, décrivant les paramètres de configuration et les étapes de réalisation d'une application CANbus.

4- Gestion avancée du protocole CANopen, décrivant la gestion et l'utilisation du dictionnaire (SDO) pour configurer les modules d'entrée/sorties, l'ajout de PDO supplémentaire à la configuration CANopen, la description de l'exemple CAN_PDO_exemple.

2.1. Introduction au protocole CANbus

Le CANbus est un protocole de communication série supportant efficacement le contrôle en temps réel de systèmes distribués. Le protocole permet aux sous-systèmes raccordés au bus d'échanger les données à tour de rôle. Le bus consiste en un simple canal bidirectionnel qui transporte les messages. Cela élimine le besoin d'utiliser une connexion point-à-point pour chaque sous-système.

2.1.1. Nœud

Un nœud est un périphérique relié au réseau et capable de communiquer selon le protocole CANbus. Des nœuds peuvent être ajoutés dans un réseau fonctionnel sans nécessiter de modifications aux nœuds existants.

2.1.2. Message et identificateur de message (COB-ID)

L'information est véhiculée sur le bus à l'aide d'un message (trame de bits) de format défini, mais de longueur variable et limitée. Chaque message a un identificateur de message unique qui définit la priorité et son contenu (COB-ID : Communication Object Identifier). Ainsi tous les nœuds d'un bus reçoivent le message, et chacun est capable d'interpréter s'il lui est destiné.

L'identificateur de message est unique dans l'ensemble du réseau puisque plusieurs systèmes entrent en concurrence pour l'accès au bus. Chaque trame débute par l'identifiant de message COB-ID pour permettre l'arbitrage du bus.

2.1.3. Arbitrage du bus

Dès que le bus est libre, n'importe quel nœud relié au réseau peut émettre un nouveau message. Lorsque plusieurs nœuds émettent un message simultanément il y a contention et corruption des données. Les systèmes qui émettent des messages font la résolution de la contention par un système d'arbitrage basé sur la priorité contenue dans l'identificateur de message. Lorsqu'une contention est observée sur le bus, les stations les moins prioritaires cessent alors d'émettre. Les messages seront retransmis par les stations lorsque le bus sera à nouveau libre.

2.1.4. Les niveaux de mise en œuvre

Le RightWON supporte les spécifications CAN 2.0 A et CAN 2.0 B de l'organisme CAN in Automation (www.can-cia.org). Les implémentations CAN conçues conformément à la spécification CAN 2.0 A ou CAN 2.0 B peuvent communiquer entre elles tant qu'elles ne font pas l'usage du format étendu.

Pour plus de détails, vous pouvez aussi consulter le site http://fr.wikipedia.org/wiki/Controller_area_network.

2.1.5. Fonctionnalités du protocole CANopen

Dans le RightWON, le pilote CANbus est utilisé pour gérer les sous-systèmes CANopen. Le CANopen est un protocole d'application de couche supérieure au protocole CANbus. Le CANopen s'occupe de traiter des détails et des fonctions spécifiques à l'implémentation du CAN. Il offre différents moyens pour accéder aux données du dictionnaire d'objet : Service Data Object et Process Data Object.

L'organisme CAN in Automation (www.can-cia.org) fournit des informations supplémentaires sur le protocole CANopen. De plus, le document suivant constitue une excellente référence pour la compréhension du protocole :

www.nikhef.nl/pub/departments/ct/po/doc/CANopen30.pdf

2.1.5.1. Dictionnaire d'objet (OD)

Un dictionnaire d'objet définit, pour chacune des entrées/sorties d'un nœud, l'information sur le format de la donnée ainsi que sur le moyen d'y accéder. On accède à une entrée du dictionnaire en définissant son index unique et son sous-index [index, sous index].

2.1.5.2. Service Data Object (SDO)

Le SDO permet l'accès aux paramètres des entrées/sorties d'un nœud inscrit dans le dictionnaire d'objet. L'esclave CANopen répond en envoyant les paramètres de la donnée ou en confirmant l'écriture dans l'OD. Les requêtes envoyées ont un identifiant avec une priorité plus basse que les messages de données reçus des nœuds.

2.1.5.3. Process Data Object (PDO)

Le PDO sert au transfert des données temps réel telles que les informations de contrôle et de statut d'un module d'E/S. Le PDO accède à la valeur de la donnée dans l'OD en définissant son [index, sous index]. La requête se fait au travers d'un message qui est reçu par l'ensemble des nœuds présents sur le bus (RPDO). Ceci permet à l'ensemble des nœuds du bus de lire au même instant la donnée demandée. Les nœuds dont le message concerne transmettent l'information demandée (TPDO) sur le bus pour être reçue par le maître du réseau.

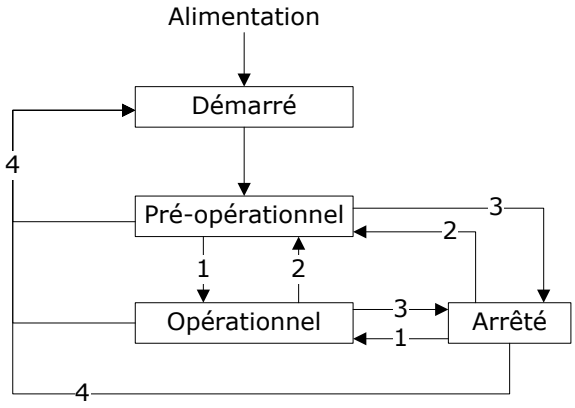
Un PDO permet de mettre un ou plusieurs champs de l'OD dans un même message court. Un PDO contient de 0 à 8 octets de données. Par exemple, un PDO peut transférer au maximum 64 données binaires ou 4 données analogiques de 2 octets.

2.1.5.4. Message NMT CANopen

Les messages NMT sont utilisés pour envoyer des commandes de changement de l'état des nœuds (ex. : démarrer ou arrêter des nœuds). Les commandes peuvent être envoyées sur un nœud ou sur tous les nœuds du bus. Tous les esclaves CANopen évaluent les commandes NMT entrantes, mais seulement un maître NMT est actif et peut envoyer des commandes NMT.

Quatre états sont possibles pour un nœud :

- Démarré: Les nœuds envoient un message Boot-up au maître du réseau pour lui indiquer que les appareils ont été enregistrés et qu'ils sont prêts à opérer. Le nœud entre ensuite automatiquement dans le mode préopérationnel.
- Préopérationnel : Dans cet état, seulement les messages SDO, les messages de contrôle d'erreur et les commandes NMT sont permis.
- Opérationnel: Tous les types de messages sont permis (NMT, SDO, PDO, contrôle d'erreur).
- Arrêté : Seule la communication par commande NMT est permise entre le nœud et le maître du réseau. Il permet aussi la surveillance des nœuds.



Les commandes NMT suivantes sont disponibles pour changer d'état :

1- **Démarrer** : Initialiser un ou des nœuds.
2- **Préopérationnel** : Le(s) nœud(s) entre dans l'état préopérationnel.
3- **Arrêter** : Arrêter la communication entre le(s) nœud(s) et le maître du réseau.
4- **Reset** : Réinitialiser les paramètres du (des) nœud(s) dans l'OD à leurs valeurs par défaut.

Voir la section « Créer un message NMT CANopen » pour plus de détails.

2.1.5.5. Contrôle d'erreurs

Le contrôle d'erreurs permet au maître d'évaluer si les esclaves CANopen fonctionnent correctement sur le réseau. Deux types de contrôle d'erreurs existent : « Heartbeat » et surveillance du nœud.

- **«Heartbeat» :** L'esclave CANopen transmet périodiquement un message signalant sa présence et son état « Hearbeat ») au maître du réseau. La période de temps est configurée dans l'OD et le maître doit recevoir au moins un message «Hearbeat» durant cette période. Le maître évalue ensuite si le nœud fonctionne correctement et s'il est dans le bon état du réseau.
- **Surveillance du nœud:** La surveillance du nœud (« Node Guarding ») permet de vérifier si le nœud fonctionne toujours dans le bon état du réseau ou non. Le maître demande à l'esclave CANopen un message de contrôle d'erreur via une trame CAN à distance (« Remote Transmit Request »). L'esclave répond avec une trame de donnée CAN qui contient son état NMT actuel.

2.1.6. Support des systèmes CANopen

Le RightWON supporte la majorité des contrôleurs CANopen offerts par les manufacturiers de systèmes d'entrées/sorties, soit entre autres :

- Phoenix Contact CANopen Bus Terminal, IL CAN BK-TC-PAC (32 x TPDO, 32xRPDO)
- Wago CANopen Fieldbus Coupler, 750-307 (5 x RPDO, 5 x TPDO) et 750-337 (32xTPDO et 32 x RPDO)
- Wieland Bus coupler unit RICOS TP BC-CANOPEN, 83.039.0120.0 (5 x RPDO, 5 x TPDO)
- Plusieurs autres produits.

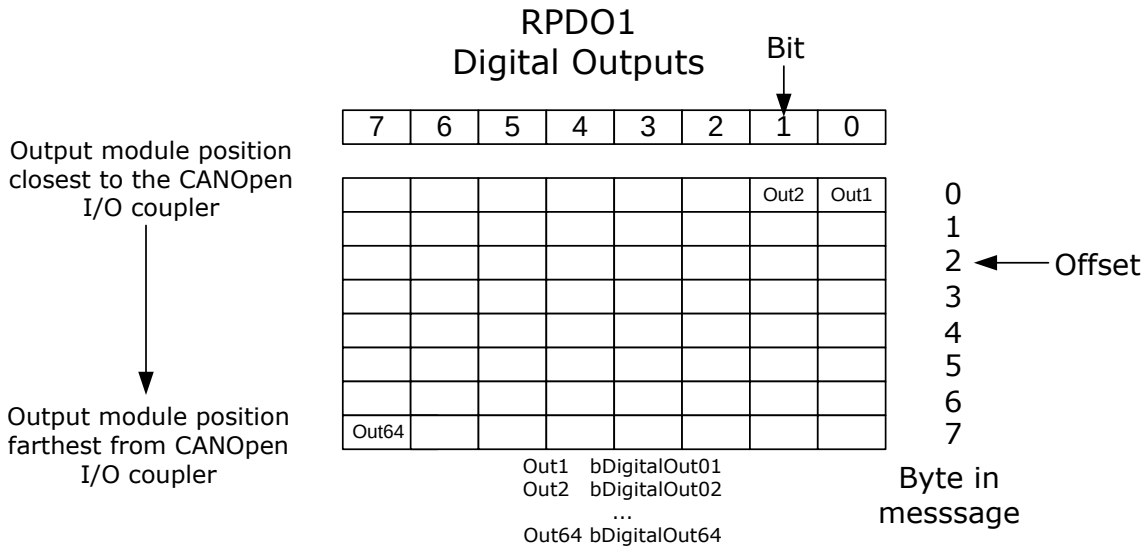
Les contrôleurs CANopen conformes à la spécification DS401 supportent par défaut (sans aucune configuration) 4 PDO pour les entrées vers le RightWON (TPDO1 à TPDO4) et 4 PDO en sortie du RightWON (RPDO1 à RPDO4), chacun comportant un maximum de 8 octets. Ces PDO sont utilisés pour le transport des données entre le RightWON et les entrées/sorties. Le TPDO1 est automatiquement assigné aux 64 premières entrées numériques trouvées à droite du contrôleur alors que le RPDO1 est automatiquement assigné aux 64 premières sorties numériques. Les TPDO2 à TPDO4 sont assignés aux premières entrées analogiques (12 entrées, soit 4 par bloc) alors que les RPDO2 à RPDO4 sont assignées aux 12 premières sorties analogiques. Il est à noter que la notion de transmission (TPDO) et de réception (RPDO) est vue du coupleur CANbus relié au RightWON.

PDO	Identificateur (hexadécimal)	Utilisation
TPDO1	181	Max. 64 entrées digitales
TPDO2	281	Max. 4 entrées (1-4) analogiques 16 bits
TPDO3	381	Max. 4 entrées (5-8) analogiques 16 bits
TPDO4	481	Max. 4 entrées (9-12) analogiques 16 bits
RPDO1	201	Max. 64 sorties digitales
RPDO2	301	Max. 4 sorties (1-4) analogiques 16 bits
RPDO3	401	Max. 4 sorties (5-8) analogiques 16 bits
RPDO4	501	Max. 4 sorties (9-12) analogiques 16 bits

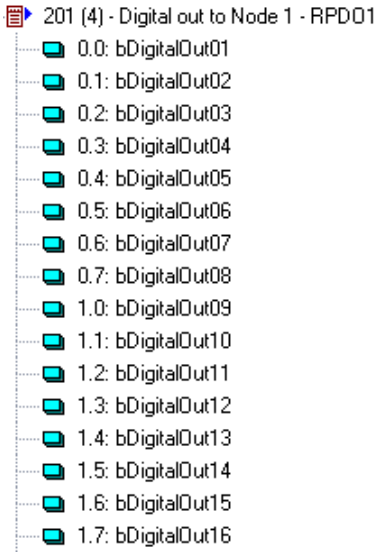
Pour plus de détails, se référer à la section «Adressage des PDO et assignation de PDO supplémentaire».

2.1.7. Fonctionnement des PDO associés aux entrées/sorties digitales

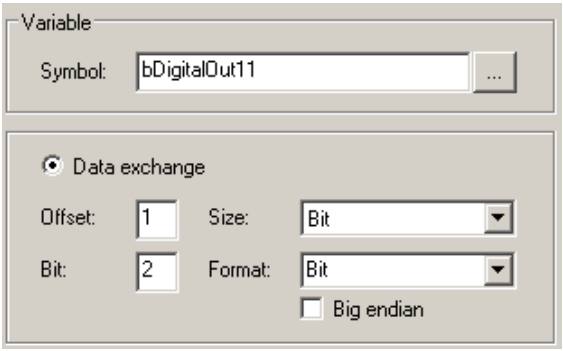
La figure suivante illustre le mapping du RPDO1 associé aux sorties numériques d'un nœud CANopen. Puisque le RPDO contient 8 octets, il peut être associé à un maximum de 64 variables booléennes. Lors de la configuration automatique d'un nœud, les sorties sont assignées dans l'ordre où elles sont retrouvées à droite du coupleur CANopen. L'assignation débute par le bit 0 à l'octet (0) jusqu'au bit 7. Lorsque plus de 8 sorties numériques sont retrouvées, elles sont assignées aux octets suivants jusqu'à ce que le bloc soit plein.



La figure suivante illustre le mapping des variables booléennes *bDigitalOut%%* aux 16 premières sorties du RPD01 possédant le COB-ID égal à 201. Le bit correspondant du RPDO est désigné par l'index *offset.bit*.



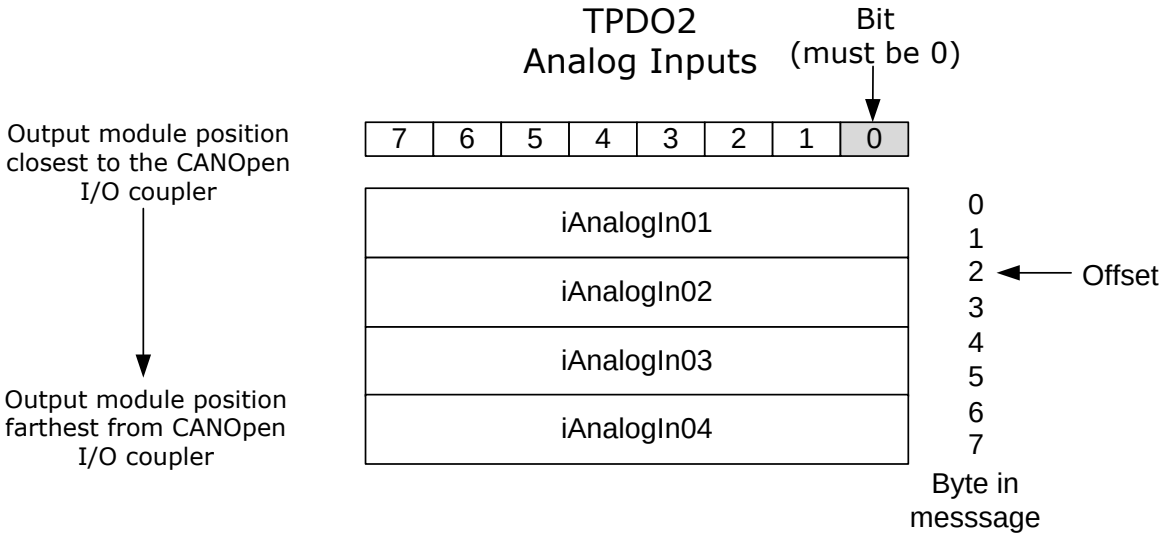
Par exemple, la variable *bDigitalOut11* est associée à l'offset 1, bit 2 (1.2) du RPD01.



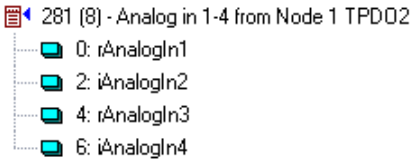
Il est à noter que le mapping des entrées numériques se fait de la même façon que celui des sorties numériques, à l'exception qu'elles sont assignées à TPDO1 plutôt qu'à RPDO1.

2.1.8. Fonctionnement des PDO associés aux entrées/sorties analogiques

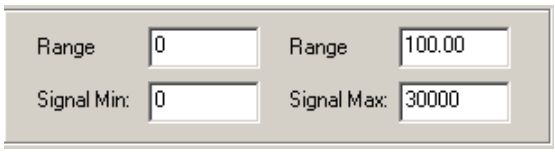
La figure suivante illustre le mapping du TPDO2 associé aux entrées analogiques d'un nœud CANopen. Puisque le TPDO contient 8 octets, il peut être associé à un maximum de 4 variables entières. Lors de la configuration automatique d'un nœud, les entrées analogiques de 12 ou 16 bits sont assignées dans l'ordre où elles sont retrouvées à droite du coupleur CANopen. L'assignation débute par les octets 0 et 1 (offset 0) puis se poursuit avec les offset 2, 4 et 6 jusqu'à ce que le bloc soit plein. Les entrées analogiques suivantes s'il y a lieu sont automatiquement assignées aux TPD03 et TPD04.



La figure suivante illustre 4 variables (nombres réels) assignées aux 4 premières entrées analogiques du coupleur CANbus du nœud 1 (TPDO2). Même si les données du TPDO sont réparties en 4 entiers signés de 16 bits, les variables de n'importe quel type peuvent être rattachées aux PDO car le pilote CANbus fait la conversion des données.



De plus, le pilote supporte la mise à l'échelle des données. Dans l'exemple suivant, une entrée analogique provenant d'un module Phoenix Contact IB IL AI 2/SF fait une conversion 4-20mA à un compte brut variant entre 0 @ 30000 (Signal Min et Max). La variable associée à l'entrée variera entre 0 et 100.00 (Range).



2.1.9. Mise à jour des sorties digitales et analogiques

Le RightWON supporte 3 méthodes pour la mise à jour des sorties digitales et analogiques:

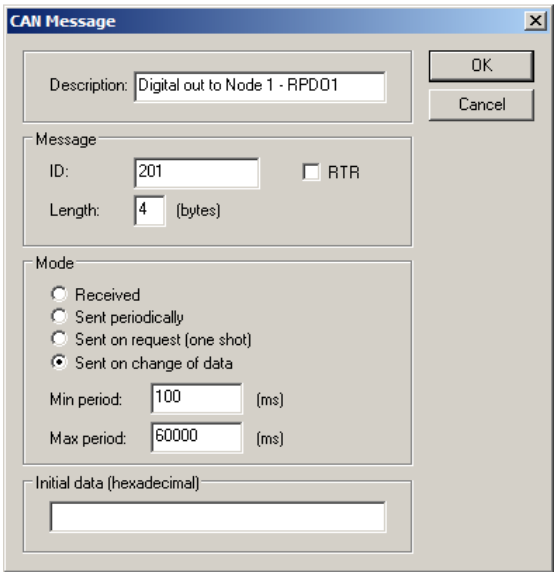
1- Changement de valeur

2- Envoi périodique

3- Envoi sur demande

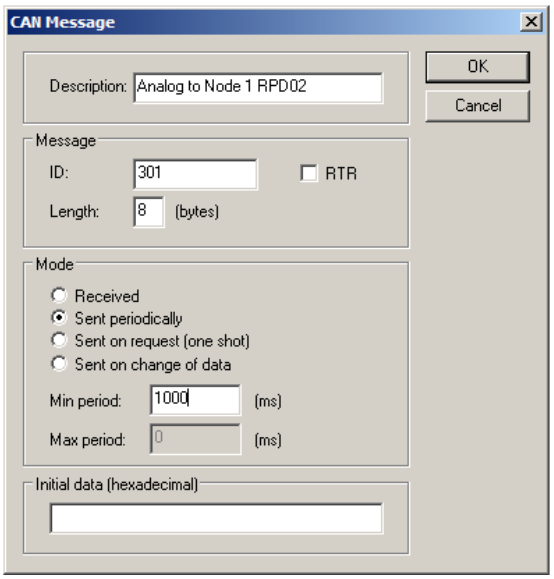
2.1.9.1. Mise à jour par changement de valeur

Dans ce mode de fonctionnement, le RPDO en entier est envoyé par le RightWON au coupleur CANopen lors du changement de l'une ou l'autre des variables assignées au RPDO. Dans le but d'empêcher la congestion du CANbus par des changements trop rapprochés, la période minimale définit une attente en milliseconde avant l'envoi du bloc lors du changement d'une valeur. La période maximale force un rafraîchissement périodique des sorties même s'il n'y a pas de changement de valeur. Cette méthode de mise à jour est couramment utilisée pour les sorties digitales.



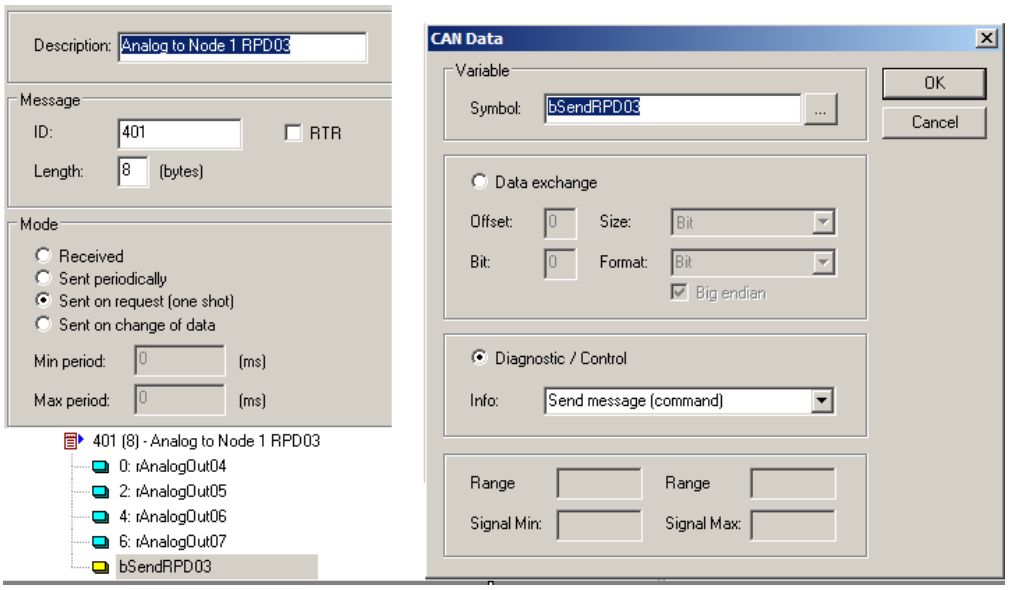
2.1.9.2. Mise à jour périodique

Dans ce mode de fonctionnement, le RPDO en entier est envoyé par le RightWON au coupleur CANopen selon la période spécifiée. C'est le mode de fonctionnement le plus couramment utilisé pour faire la mise à jour de sorties analogiques.



2.1.9.3. Mise à jour sur demande

Dans ce mode de fonctionnement, le RPDO en entier est envoyé par le RightWON lorsque la requête associée devient active. Le pilote remet la requête à 0 une fois traitée. Cette requête est associée au bloc de données en la configurant à un échange de diagnostic/contrôle plutôt qu'un échange de données.



2.1.10. Mise à jour des variables à partir des entrées digitales et analogiques

Le RightWON supporte 3 méthodes pour la mise à jour des variables à partir des entrées digitales et analogiques :

1- Par événement

2- Par requête RTR

3- Par requête SYNC

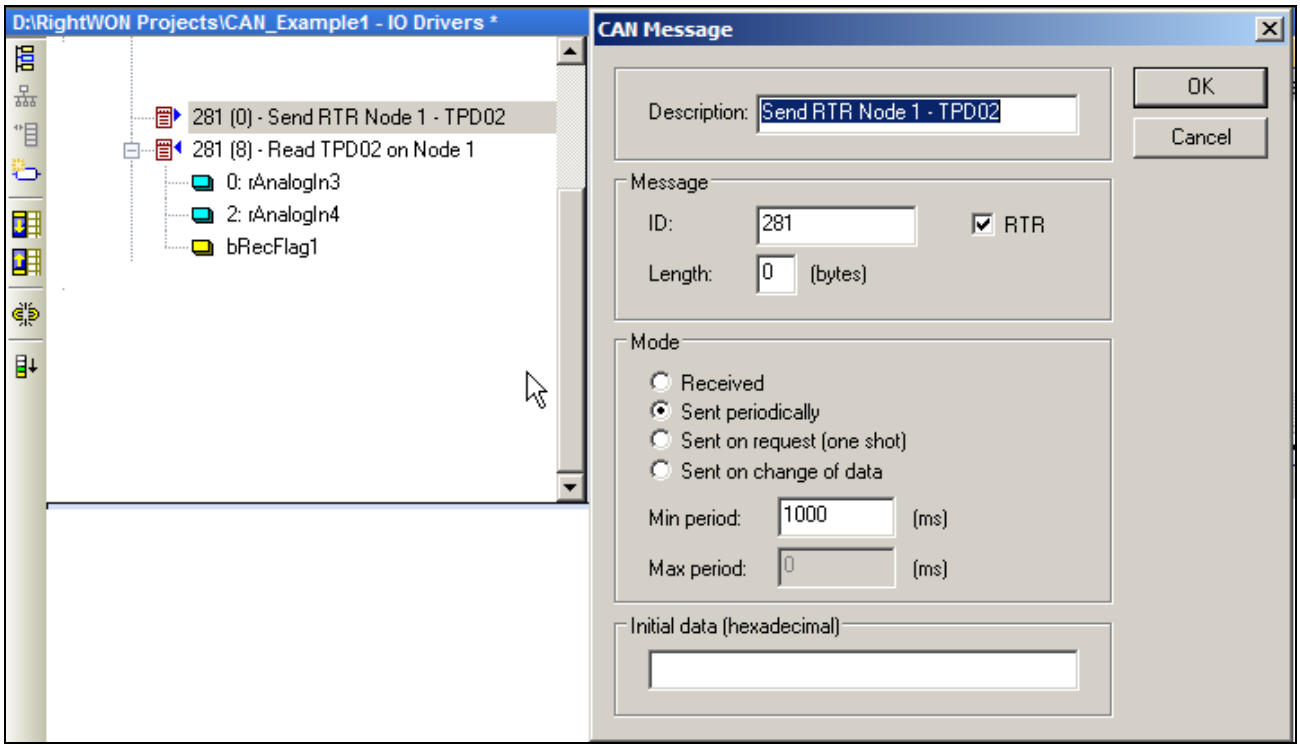
2.1.10.1. Mise à jour par événement

La mise à jour par événement est la méthode par défaut utilisée pour les entrées numériques. À chaque fois que le coupleur CANopen détecte un changement sur une des entrées associées au TPDO1, le bloc est envoyé au contrôleur CAN. Il est à noter qu'un nombre important d'événements peut surcharger le bus de communication.

Le mode de fonctionnement par événement est configurable bit à bit pour chacune des entrées digitales du système. De plus, le mode de fonctionnement par événements peut aussi être utilisé pour les entrées analogiques en définissant des bandes mortes et/ou des limites. La configuration de ces modes de fonctionnement nécessite une programmation avancée en utilisant des requêtes SDO ou en réalisant un paramétrage à l'aide d'un configurateur CANopen (voir la section « Configuration des contrôleurs CANopen »).

2.1.10.2. Mise à jour par requête RTR

La mise à jour par requête RTR (Remote Transmission Request) est la méthode la plus utilisée pour lire les entrées analogiques. Il s'agit d'une requête d'interrogation générée par le RightWON au coupleur CANopen afin que celui-ci envoie le TPDO correspondant. Il s'agit en quelque sorte d'une requête de polling. Tout comme la mise à jour des sorties, cette requête peut être configurée comme un envoi périodique ou sur demande.



2.1.10.3. Mise à jour par requête SYNC

La mise à jour par requête SYNC permet l'envoi des données en synchronisation avec ce message initié par le RightWON. Un seul message SYNC peut engendrer l'envoi automatique et séquentiel de tous les objets TPDO par le coupleur ou l'échantillonnage simultané des entrées. La configuration de ce mode de fonctionnement nécessite une programmation avancée en utilisant des requêtes SDO ou en réalisant un paramétrage à l'aide d'un configurateur CANopen (voir la section « Configuration des contrôleurs CANopen »).

2.1.11. Configuration des contrôleurs CANopen

Les configurations d'entrées/sorties étendues qui dépassent 64 entrées numériques (TPDO1), 64 sorties numériques (RPDO1), 12 entrées analogiques (TPDO2 à TPDO4) ou 12 sorties analogiques (RPDO2 à RPDO4) nécessitent la configuration du contrôleur CANopen. Ce paramétrage permet de faire l'assignation des modules d'entrée/sorties aux PDO. Plusieurs logiciels de configuration CANopen sont disponibles auprès des fournisseurs de matériel d'entrées/sorties. Cependant, certains éditeurs spécialisés offrent des logiciels de configuration et de vérification CANopen. À cet effet, les logiciels *CANopen Configuration Studio* (http://www.ixxat.com/canopen_configurationstudio_en.html) et *CANopen Magic Pro* (http://www.canopenmagic.com/peak/professional.htm) constituent des excellents outils de travail.

2.1.12. Adressage des PDO et assignation de PDO supplémentaire à un nœud

Selon le protocole CANopen et la spécification DS401, l'identificateur de message COB-ID contient 4 bits de codes de fonction pour désigner les types d'objet (NMT, TPDO1, RPDO3, TSDO, etc.) et 7 bits d'adresse pour désigner un nœud. L'adresse 0 désigne tous les nœuds.

L'identificateur des PDO est composé de l'adresse de base additionnée à l'adresse du nœud. Les adresses de base des PDO du nœud 1 sont illustrées dans le tableau suivant. Selon ce tableau, le TPDO1 du nœud 3 sera donc égal à 183 et le RPDO4 du nœud 5 sera égal à 505.

PDO	Identificateur (hexadécimal)	Utilisation
TPDO1	181	Max. 64 entrées digitales
TPDO2	281	Max. 4 entrées (1-4) analogiques 16 bits
TPDO3	381	Max. 4 entrées (5-8) analogiques 16 bits
TPDO4	481	Max. 4 entrées (9-12) analogiques 16 bits
RPDO1	201	Max. 64 sorties digitales
RPDO2	301	Max. 4 sorties (1-4) analogiques 16 bits
RPDO3	401	Max. 4 sorties (5-8) analogiques 16 bits
RPDO4	501	Max. 4 sorties (9-12) analogiques 16 bits

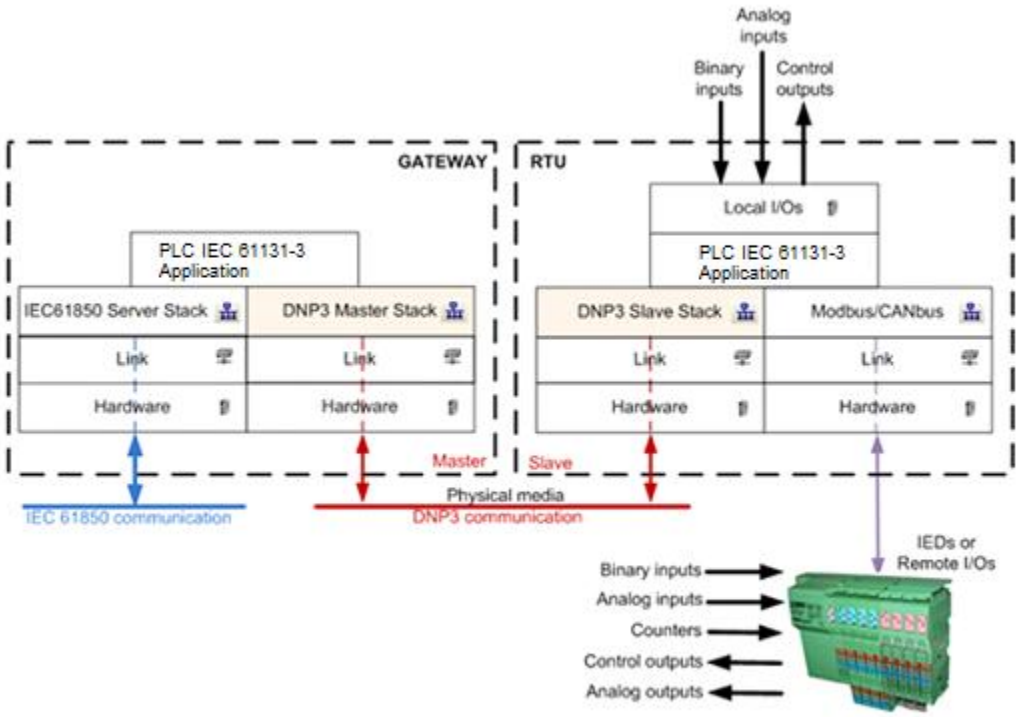
Le protocole de base définit uniquement les PDO 1 à 4 pour désigner les objets de données. Par conséquent, les données supplémentaires doivent être assignées par le configurateur CANopen à des identificateurs non utilisés sur le réseau (par exemple les PDO d'un autre nœud non installé). Voici un exemple d'assignation de PDO supplémentaire pour le nœud 1 en utilisant les PDO non installés du nœud 2 :

PDO	Identificateur (hexadécimal)	Utilisation
TPDO5	182	Max. 64 entrées digitales
TPDO6	282	Max. 4 entrées (1-4) analogiques 16 bits
TPDO7	382	Max. 4 entrées (5-8) analogiques 16 bits
TPDO8	482	Max. 4 entrées (9-12) analogiques 16 bits
RPDO5	202	Max. 64 sorties digitales
RPDO6	302	Max. 4 sorties (1-4) analogiques 16 bits
RPDO7	402	Max. 4 sorties (5-8) analogiques 16 bits
RPDO8	502	Max. 4 sorties (9-12) analogiques 16 bits

La section « Ajout de PDO supplémentaire à la configuration CANopen » décrit comment configurer des PDO pour être ajoutés à un nœud (coupleur CANopen).

Intégration du protocole CANbus dans le RightWON

Chaque unité RightWON CPU (RWU 010000) comporte un port CANbus. Le mode de fonctionnement du CANbus est illustré dans une application typique à la figure suivante. Toutes les données véhiculées par le protocole CANbus se traduisent par variables dans l'applicatif PLC IEC 61131-3. Ainsi, les protocoles sont gérés à partir du gestionnaire de bus de terrain sous la supervision de programmes d'automatisation PLC. Le gestionnaire de bus de terrain échange les variables entre le pilote CANbus et l'applicatif PLC.



Les unités RightWON supportent le protocole CANopen par le pilote CANbus paramétré avec le configurateur Drivers E/S. Le lien logique entre le port physique offert par la configuration matérielle du RightWON et le pilote est géré par le lien de communication CANbus. Ce dernier est paramétré par le gestionnaire de Liens dans le Configurateur réseau.

3.1. Configuration des E/S CANopen

Les variables binaires et analogiques du RightWON contiennent les données à échanger avec les modules d'un nœud CANbus. Par l'entremise de la configuration du pilote CANbus, les variables sont assignées aux RPDO et TPDO.

Par ordre hiérarchique, l'intégrateur doit définir :

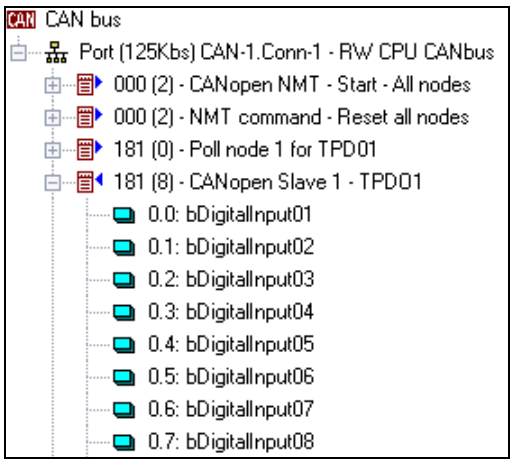
1- Le lien de communication CANbus avec le gestionnaire réseau.

2- La configuration CANbus qui intègre le pilote CANbus à l'application chargée dans le RightWON.

3- Le port CANbus utilisé pour établir la communication avec les dispositifs raccordés au RightWON. C'est à ce niveau qu'est défini le débit du réseau.

4- Les requêtes d'échange de données, de commandes et de contrôle avec les nœuds

5- S'il y a lieu, les variables associées aux échanges de données et de contrôle à plus bas niveau.



Consulter la section « Tutoriel de configuration du protocole CANbus » pour la mise en œuvre du CANbus/CANopen.

Tutoriel de configuration du CANbus - CANopen

Pour configurer le CANbus, vous devez exécuter les étapes suivantes :

1- Créer un nouveau projet RightWON
2- Ajouter et configurer un lien CANbus dans le configurateur réseau.
3- Ajouter le pilote du protocole CANbus à la configuration du RightWON.
4- Insérer et configurer un port.
5- Créer les TPDO et RPDO pour l'échange des données entre le RightWON et le coupleur CANopen
6- Configurer le mode d'échange des PDO entre le RightWON et le coupleur CANopen
7- Créer les requêtes RTR périodiques pour rapatrier les entrées analogiques et numériques
8- Insérer les variables dans les PDO
9- Insérer et configurer un message NMT CANopen.
10- Démarrer les nœuds par le programme pStartup

Le projet CAN_Tutorial contient le programme réalisé en fonction d'un coupleur CANopen comportant 16 entrées/sorties digitales et 4 entrées/sorties analogiques.

4.1. Créer un nouveau projet RightWON

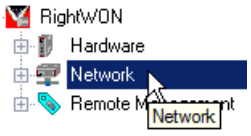
Pour créer un nouveau projet RightWON, veuillez vous référer au manuel « RWM000080-MA-fr, RightWON Configuration Suite – Guide d'application ». Le projet RightWON doit comporter au minimum la configuration RightWON dans le configurateur de bus de terrain.

4.2. Ajouter et configurer un lien CANbus

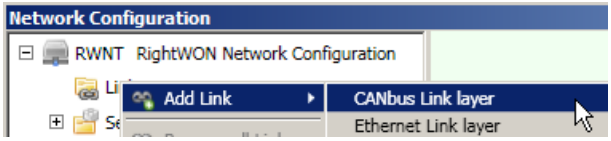
Afin de pouvoir utiliser les modules E/S CANopen, il faut ajouter un lien CANbus pour coupler le matériel et le pilote CANbus.

Consulter le manuel « RWM000010-MA-fr, RightWON Configuration Suite – Manuel » pour plus de détails sur le configurateur réseau RightWON.

1- Dans la configuration **RightWON** accédée dans fenêtre Drivers E/S, double-cliquer sur **Network**.

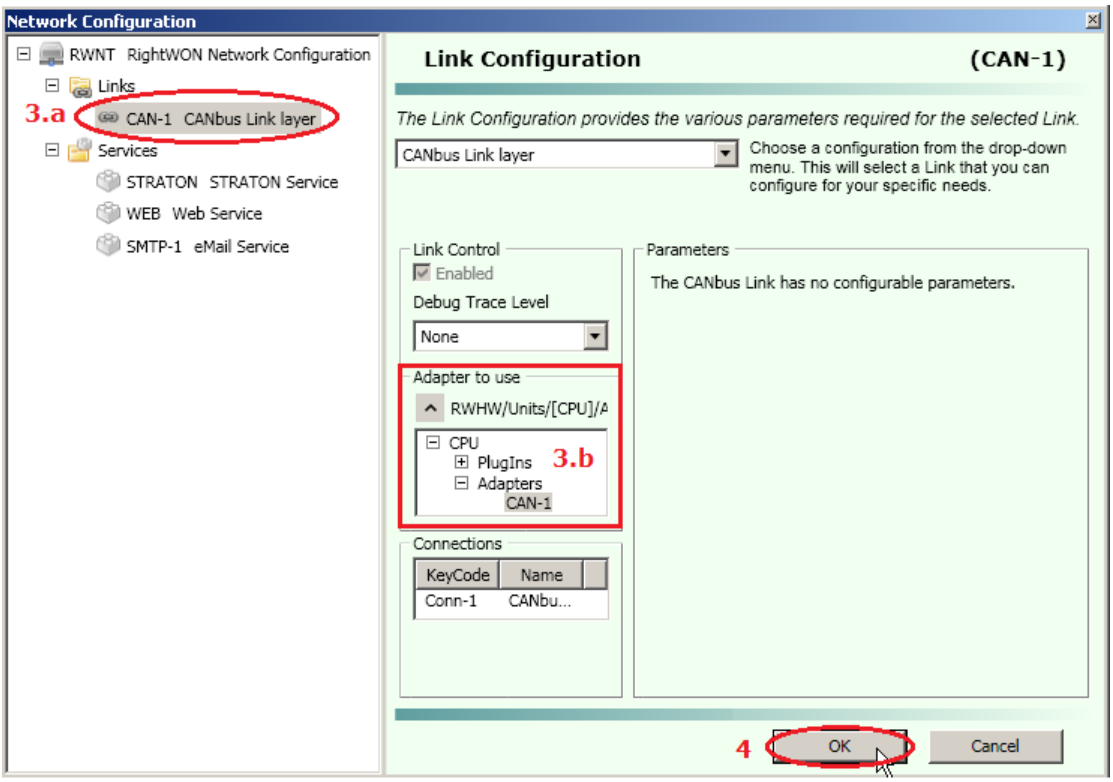


2- Dans le configurateur réseau, cliquer avec le bouton de droite de la souris sur **Links**. Sélectionner **Add Link** et cliquer sur **CANbus Link Layer**.



3- Si le projet comporte uniquement une CPU sans MCU, le couplage entre le lien CANbus et le matériel se fait automatiquement. Si le projet comporte une CPU avec une ou deux MCUs, le lien CAN doit être couplé avec l'adaptateur de la CPU ou de la MCU utilisé pour la communication avec les IEDs CANbus. Pour ce faire, exécuter les étapes suivantes :
   a. Dans la zone de navigation, Sélectionner **CAN-1** dans la section **Links**.
   b. Ensuite, dans la zone de travail, sélectionner l'adaptateur **CAN-1** dans la section **Adapter to use.**

4- **Cliquer** sur **OK**.



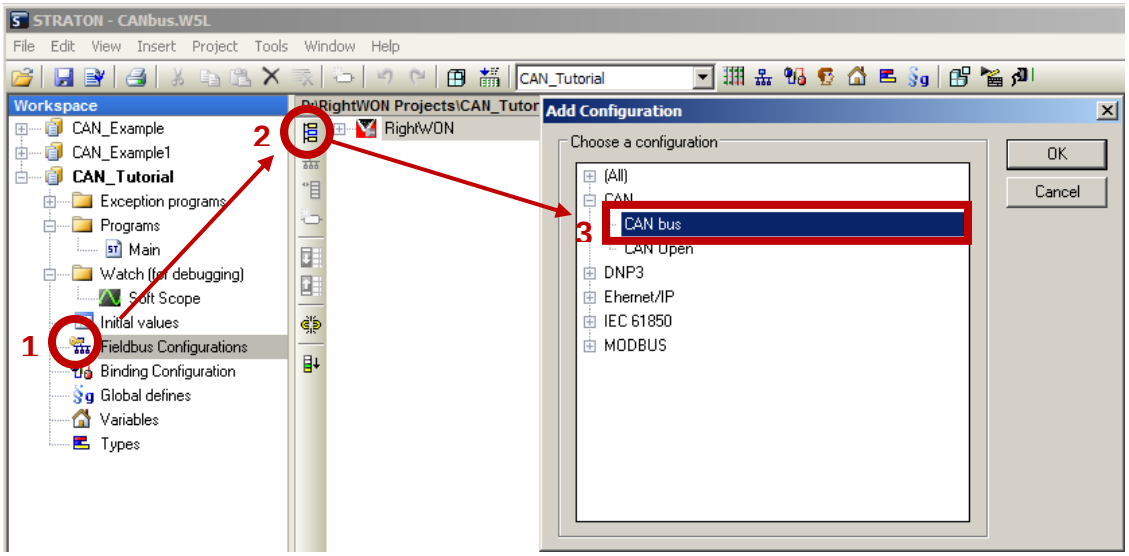
4.3. Ajouter et paramétrer le pilote du protocole CANbus

Pour ajouter le protocole CANbus à la configuration RightWON, suivre les étapes suivantes :

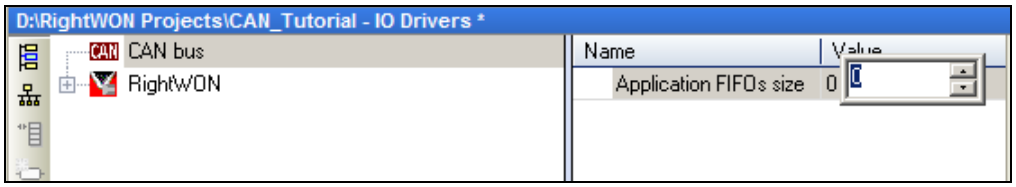
1- Dans l'espace de travail, double-cliquer sur **Configurations de bus terrain**.

2- Dans la fenêtre Drivers E-S, cliquer sur l'icône **Insérer une configuration**.

3- Sélectionner **CANbus** dans la liste et cliquer ensuite sur **OK**.



4- Dans la fenêtre **Drivers E/S**, double cliquer sur **CANbus** pour éditer les paramètres associés au pilote (voir la section « Paramétrage du pilote CANbus »). Cette configuration n'est pas requise dans le tutoriel.

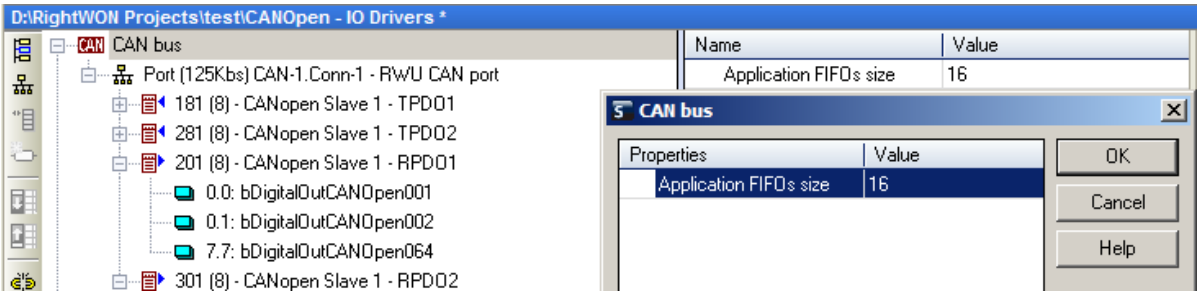


5- Après avoir ajouté et configuré le pilote CANbus, il faut ensuite ajouter et configurer le port physique CANbus.

4.3.1. Paramétrage du pilote CANbus

Le paramètre « **Application FIFOs size** » doit être ajusté en fonction d'un minimum du double du nombre de requêtes simultanées définies par programmation à l'aide des fonctions CanRcvMsg et CanSndMsg lorsqu'elles sont utilisées. Ce FIFO (pile premier entré, premier sorti) gère le stockage des requêtes générées par l'application et celles envoyées sur le bus. Par exemple, au cours d'un même cycle automate, plusieurs requêtes pourraient être générées simultanément et sont stockées dans le FIFO jusqu'à ce que le bus se libère et soit prêt à échanger un nouveau message. Plus le nombre de requêtes est grand et plus les échanges sont rapides, plus grand doit être défini le FIFO.

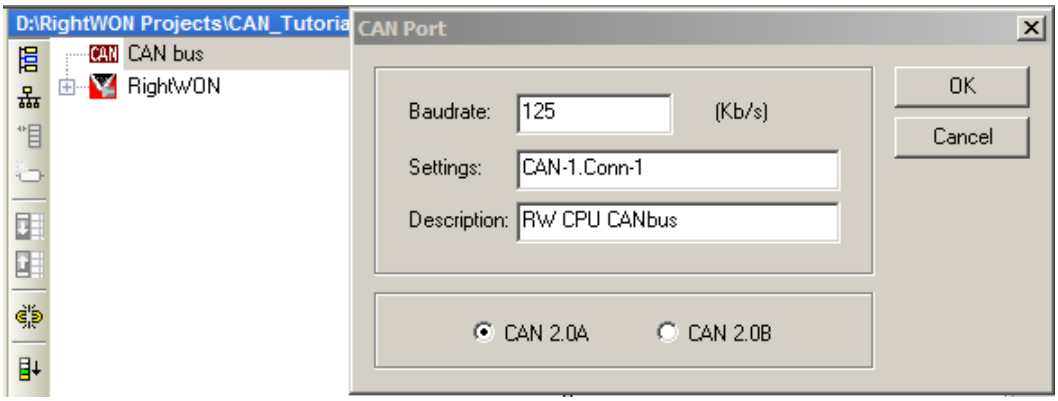
Lorsque le pilote CANbus est entièrement géré par le configurateur de bus de terrain, le FIFO n'est pas requis et peut être laissé à 0.



Lors de la mise en route, la cause fréquente de la perte de données est le débordement du FIFO ou le débit trop important de messages par rapport à la capacité du réseau. Augmenter la dimension du FIFO règle le problème dans plusieurs cas.

4.4. Insérer et paramétrer le port CANbus

1- Dans la zone **Drivers E/S**, sélectionner **CAN bus** et activer la commande **Insérer un maître/un port...**



2- Une fenêtre de configuration du Port CAN s'ouvre. Saisir les données suivantes:

Baudrate: Débit de communication auquel est opéré le CANbus. Tous les nœuds d'un réseau doivent communiquer au même débit. Pour la majorité des contrôleurs CANbus, ce débit est paramétré par des micro-interrupteurs. Le débit est fonction de la longueur du réseau et du type de câble selon le tableau suivant. Un câble blindé à paires torsadées d'impédances variant entre 108 et 132 Ohms est recommandé.

Débit	Longueur
1000Kb/s	30m
800Kb/s	50m
500Kb/s	100m
250Kb/s	250m
125Kb/s	500m

Settings: Inscrire l'adresse du lien de communication CAN défini dans le configurateur réseau soit habituellement *CAN-1.*Conn-*1* qui désigne le lien associé au contrôleur CANbus intégré à la CPU RWU 010000 (dans le cas où il n'y a pas de MCU). Voir la section « Ajouter et configurer un lien CANbus » pour plus de détails.

Description : Entrer le nom d'une connexion significative à votre application.

CAN 2.0A/CAN 2.0B: Entrer le niveau de compatibilité du protocole requis avec les IEDs raccordés au bus.

- Le CAN standard ou CAN 2.0 A : avec un identifiant d'objet codé sur 11 bits, qui permet d'accepter théoriquement jusqu'à 2 048 types de messages (limité à 2 031 pour des raisons historiques).
- Le CAN étendu ou CAN 2.0 B : avec un identifiant d'objet codé sur 29 bits, qui permet d'accepter théoriquement jusqu'à 536 870 912 types de messages. À la demande du SAE qui est à l'origine du standard J1939.

Vérifier la compatibilité avec le fabricant d'E/S. En cas de doute, choisir le protocole CAN 2.0A.

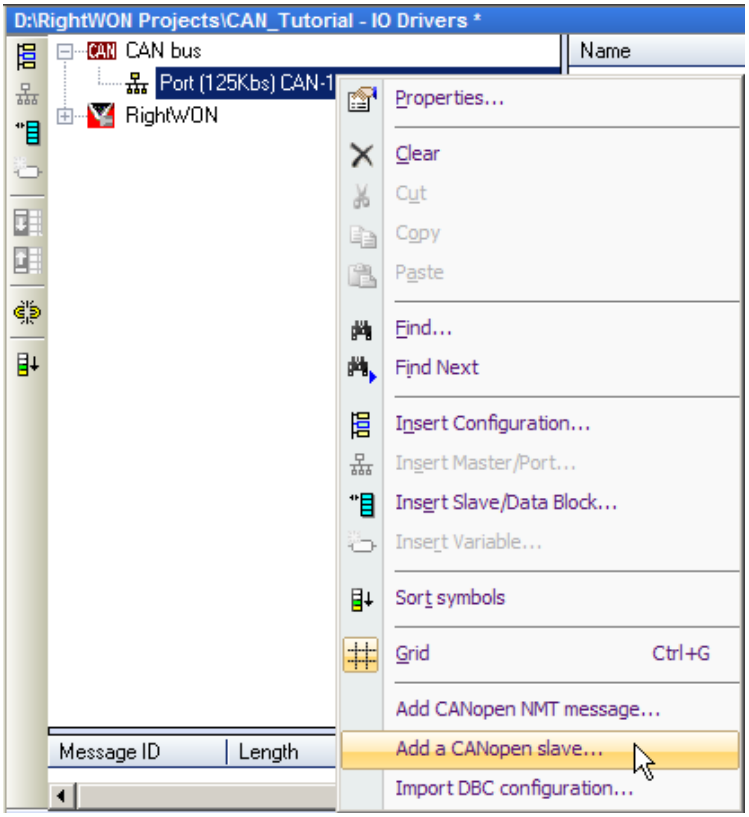
3- Pour créer le port, cliquer sur **OK**. Une fois le port ajouté, il faut ensuite créer les PDO nécessaires aux échanges de données avec le coupleur CANopen.

4.5. Créer les TPDO et RPDO pour l'échange des données

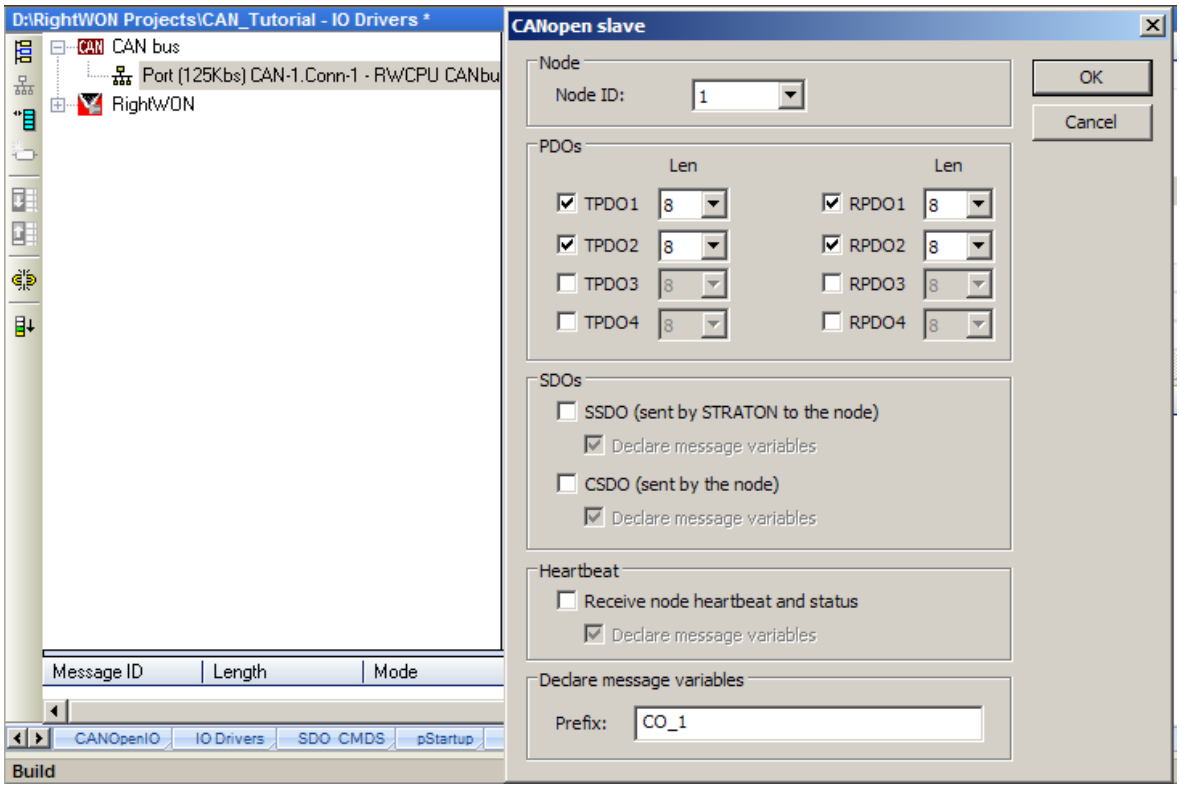
Le menu **Ajouter un esclave CANopen...** permet de définir les PDO, les SDO et les messages Heartbeat échangés entre le RightWON et le coupleur CANopen.

Les étapes suivantes créent les TPDO (Coupleur → RightWON) et les RPDO (RightWON → Coupleur) requis pour échanger les données relatives aux entrées/sorties.

1- Dans la zone de navigation, cliquer sur **Configurations de bus terrain** pour accéder à la zone **Drivers E/S**. Sélectionner le **Port CANbus** et faire un clic droit pour sélectionner **Ajouter un esclave CANopen...**



2- La fenêtre de configuration de l'esclave CANopen s'ouvre. Il faut alors configurer les paramètres selon les informations suivantes.



Dans la section **NODE** (nœud) :

Node ID : Choisir l'adresse du nœud en fonction de votre coupleur CANopen (1 à 127. L'adresse 0 désigne tous les nœuds, il ne faut pas l'utiliser).

Dans la section **PDO** (« Process Data Objects »), cocher les PDO requis pour l'échange de données:

☑ **TPDO1 :** PDO associé à un maximum de 64 entrées numériques du coupleur CANopen.

☑ **TPDO2 :** PDO associé aux entrées analogiques 1 à 4 du coupleur CANopen.

☐ **TPDO3 :** PDO associé aux entrées analogiques 5 à 8 du coupleur CANopen.

☐ **TPDO4 :** PDO associé aux entrées analogiques 9 à 12 du coupleur CANopen.

☑ **RPDO1 :** PDO associé à un maximum de 64 sorties numériques du coupleur CANopen.

☑ **RPDO2 :** PDO associé aux sorties analogiques 1 à 4 du coupleur CANopen.

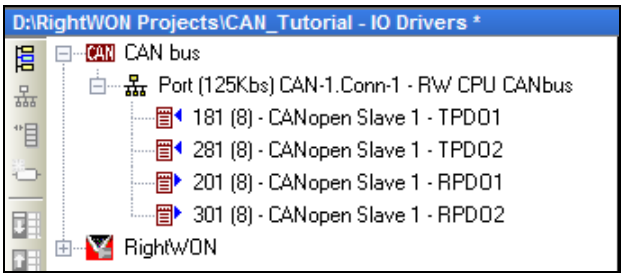
☐ **RPDO3 :** PDO associé aux sorties analogiques 5 à 8 du coupleur CANopen.

☐ **RPDO4 :** PDO associé aux sorties analogiques 9 à 12 du coupleur CANopen.

Pour chaque PDO :

LEN : Longueur du message entre 1 et 8 octets. Une longueur de 8 octets permet de transporter 64 entrées/sorties numériques ou 4 entrées/sorties analogiques 16-bit.

3- Pour créer les PDO, cliquer sur **OK**. Vous devriez créer les PDO nécessaires pour faire l'acquisition des entrées digitales et analogiques et la mise à jour des sorties digitales et analogiques. Après ces étapes, le configurateur ressemble au contenu de la figure suivante.



4- Après avoir créé les PDO, il faut ensuite configurer les modes d'échange des données entre le coupleur CANopen et le RightWON.

4.6. Configurer les modes d'échange des PDO

Une fois les PDO créés, il faut paramétrer comment ils sont échangés avec le coupleur CANopen :

- Sorties digitales (RPD01)
- Sorties analogiques (RPD02 à RPD04)
- Entrées digitales (TPD01)
- Entrées analogiques (TPD02 à TPD04)

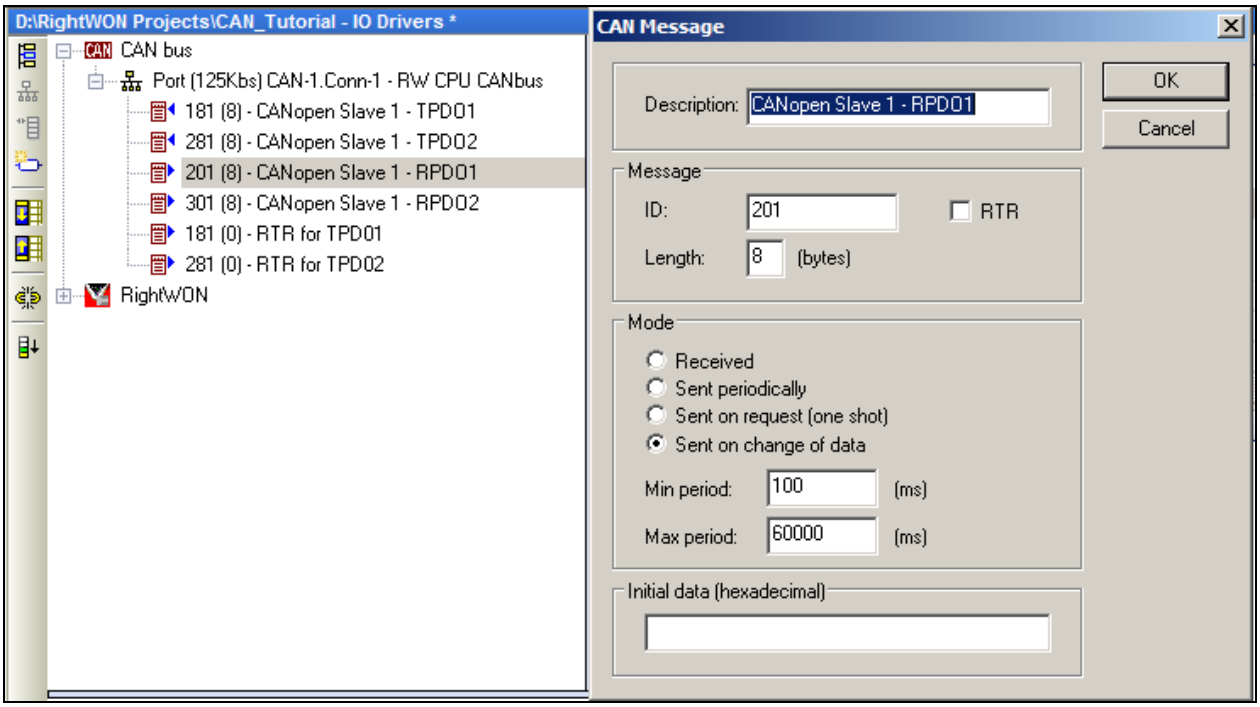
Pour plus de détails, voir les sections « Mise à jour des sorties digitales et analogiques » et « Mise à jour des variables à partir des entrées digitales et analogiques ».

Pour paramétrer les PDO, double cliquer sur le PDO à configurer dans la fenêtre **Drivers E/S**. Configurer la fenêtre qui apparaît selon les sections suivantes.

Pour faire l'acquisition des entrées analogiques et digitales, il faudra en plus créer des requêtes RTR périodiques pour rapatrier les informations périodiquement au RightWON.

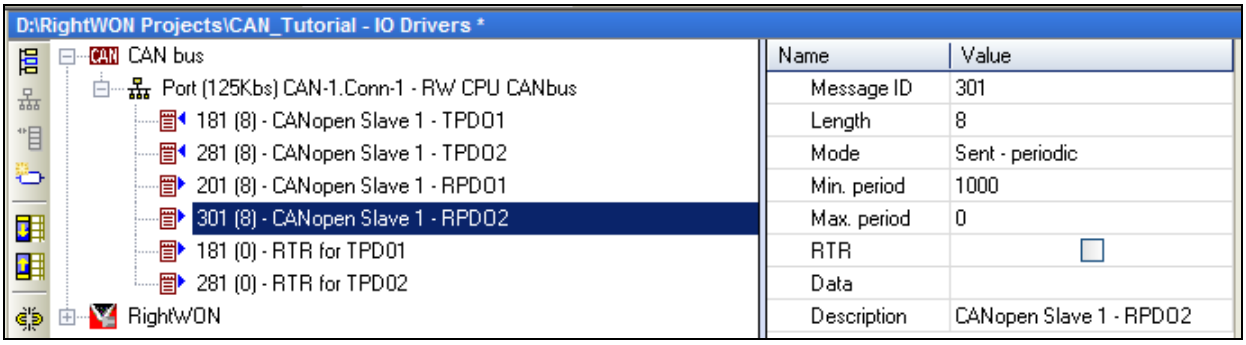
4.6.1. Sorties digitales

Les sorties digitales (RPD01) sont habituellement envoyées au coupleur sur changement d'état. Configurer la période minimale à 100ms (pour empêcher de saturer le CANbus dès le changement successif de plusieurs sorties) et la période maximale à 1 minute (en cas où il y aurait des problèmes).



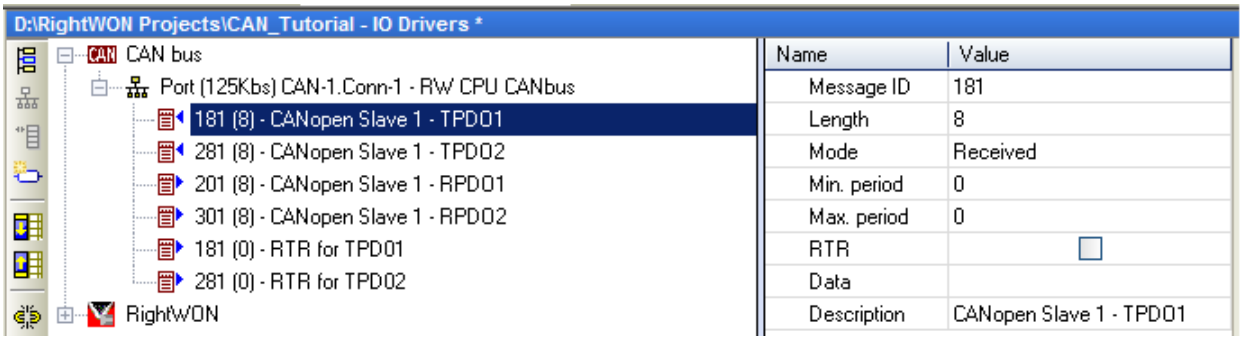
4.6.2. Sorties analogiques

Les sorties analogiques (RPDO2 à RPDO4) sont habituellement mises à jour périodiquement (par exemple à la seconde).



4.6.3. Entrées digitales

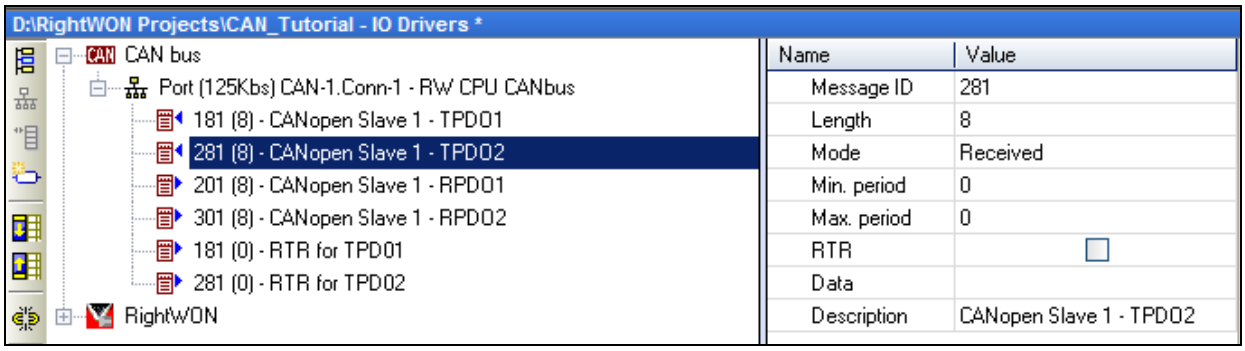
Les entrées digitales (TPDO1) sont mises à jour par événement, et donc rapatriées automatiquement par le coupleur au RightWON. Le bloc TPD01 est utilisé pour recevoir les données et ne nécessite aucun paramétrage.



Par contre, il est toutefois recommandé de faire une requête RTR périodique pour rapatrier par exemple le bloc à chaque minute au cas où des événements auraient été perdus.

4.6.4. Entrées analogiques

Les entrées analogiques (TPDO2 à TPDO4) doivent être rapatriées avec une requête RTR car par défaut les contrôleurs CANopen ne génèrent pas d'événements.



4.7. Création des requêtes RTR périodiques

Les requêtes RTR sont utilisées pour forcer l'envoi au RightWON des TPDO par le coupleur CANopen. Les requêtes peuvent être périodiques ou déclenchées sur demande. Il faut créer une requête pour chaque TPDO à rapatrier au RightWON. Les requêtes périodiques sont créées de la manière suivante :

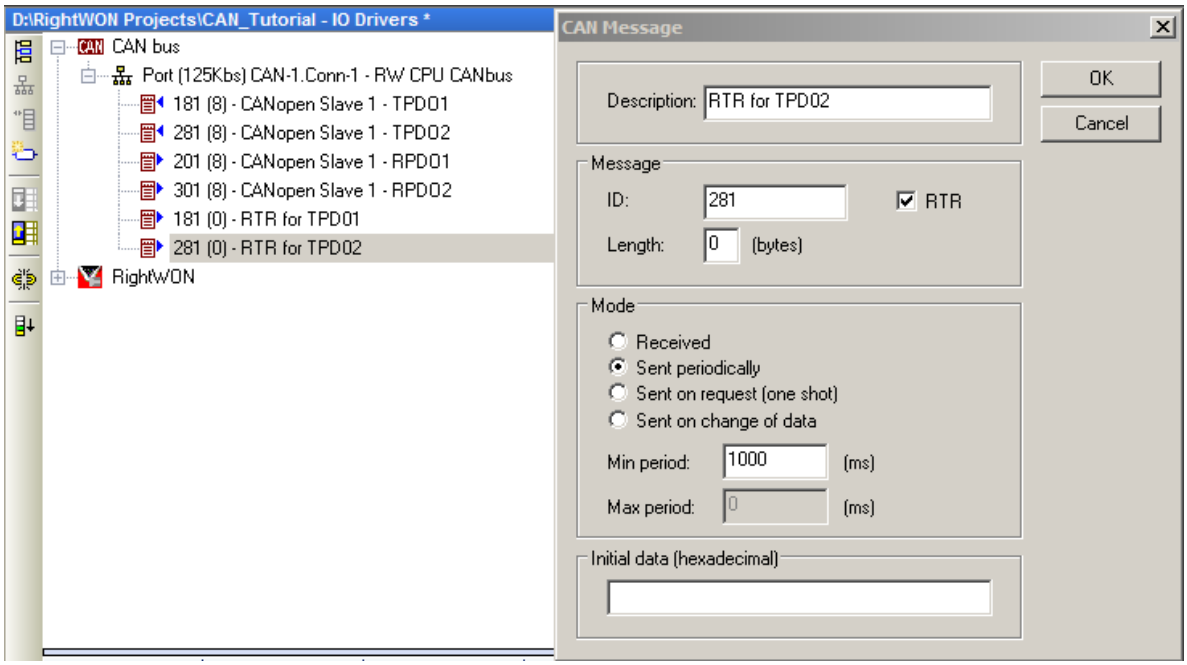
1- Sélectionner le Port CANbus dans la fenêtre I/O Drivers puis cliquer sur **Insert Slave/Data Block** .

2- Entrer le champ de la description de la requête selon vos préférences

3- Entrer le numéro d'identification du TPDO (181 pour le TPDO1 du nœud 1, 281 pour le TPDO2 du nœud 1, 182 pour le TPDO 1 du nœud 2, etc.) et une longueur de 0 puisqu'il n'y a aucune donnée à transporter. La requête déclenchera l'envoi du TPDO par le coupleur dès sa réception.

4- Cocher **RTR** pour signifier qu'il s'agit d'une requête d'envoi.

5- Sélectionner l'envoi périodique (60 000 ms recommandé pour le TPDO1 et 1 000 ms pour les requêtes TPDO2 à TPDO4).



⚠ Pour certains coupleurs CANopen (exemple : WAGO 750-337 et 750-338), la transmission des TPDO est désactivée dans la configuration par défaut. C'est-à-dire que les données des entrées analogiques ne sont lues qu'une seule fois et par la suite, ne sont pas mises à jour. Pour être en mesure d'utiliser ces données via les PDO, il faut configurer ce mode en utilisant des requêtes SDO, se référer à l'exemple d'utilisation des requêtes SDO pour l'initialisation.

4.8. Insérer une variable dans un PDO

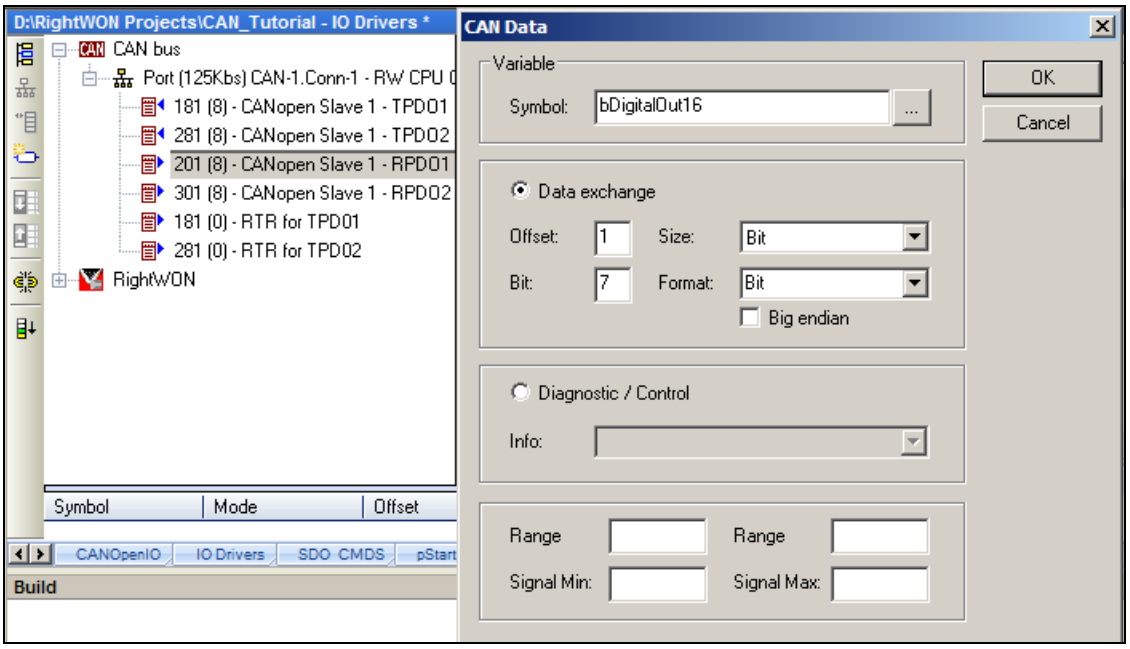
Les entrées/sorties du coupleur CANopen sont échangées avec des variables par l'entremise des PDO. Les TPDO sont utilisés pour les entrées et les RPDO pour les sorties.

4.8.1. Associer les variables aux entrées/sorties digitales

Les entrées digitales sont associées au TPDO1 alors que les sorties numériques sont associées au RPDO1. Pour insérer une variable dans un PDO, suivre les étapes suivantes :

1- Dans la zone **Drivers E/S,** cliquer avec le bouton de droite de la souris sur **CANopen Slave** et sélectionner **Insérer une variable...**

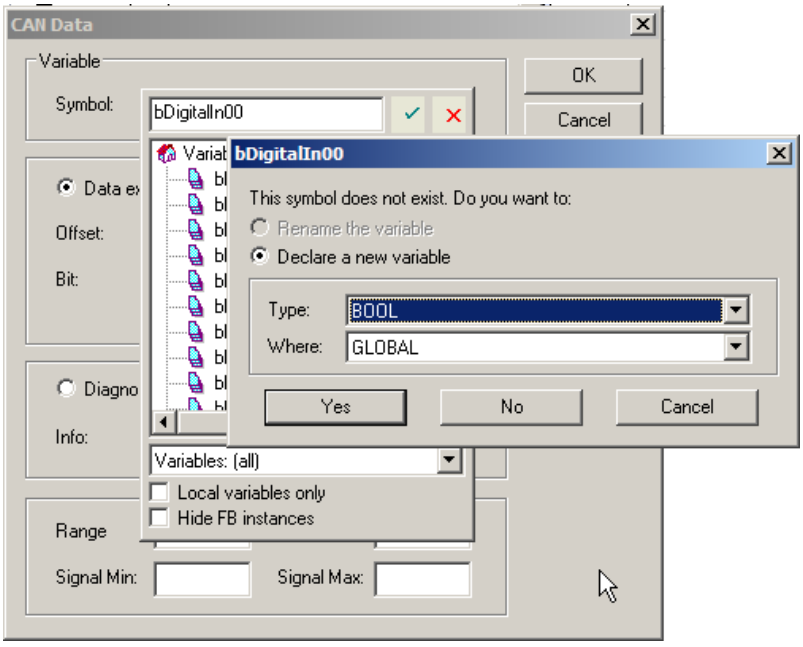
2- La fenêtre de configuration des données du PDO s'ouvre. Cette fenêtre permet d'assigner jusqu'à 64 variables au PDO en utilisant un système de référence *offset.bit,* où l'*offset* désigne l'octet 0 à 7 du PDO et le *bit* désigne le bit dans l'octet (0 = moins significatif). Pour plus de détails, voir la section « Fonctionnement des PDO associés aux entrées/sorties digitales ».



3- Assigner les champs suivants :

Symbole: Nom de la variable associée à l'entrée/sortie. Le bouton ... permet d'insérer une variable :

- Sélectionner une variable dans la liste et cliquer sur √.
- Ou créer une nouvelle variable en inscrivant son nom et en cliquant sur √.
  Note : De préférence donnez-vous une convention de nomenclature des variables. Par exemple, préfixer les variables de type DINT par *di*, les INT par un *i* ou les BOOL par un *b.*
  Une fenêtre de configuration de la variable s'ouvre. Choisir le **Type** booléen et assigner comme une variable globale.



Data exchange : Cocher pour l'échange des données.

Offset : Spécifier l'offset d'octet dans le PDO (0 à 7).

Bit : Spécifier le numéro de bit dans l'octet (0 à 7).

Size : Puisqu'il s'agit du TPDO1/RPDO1, spécifier un bit.

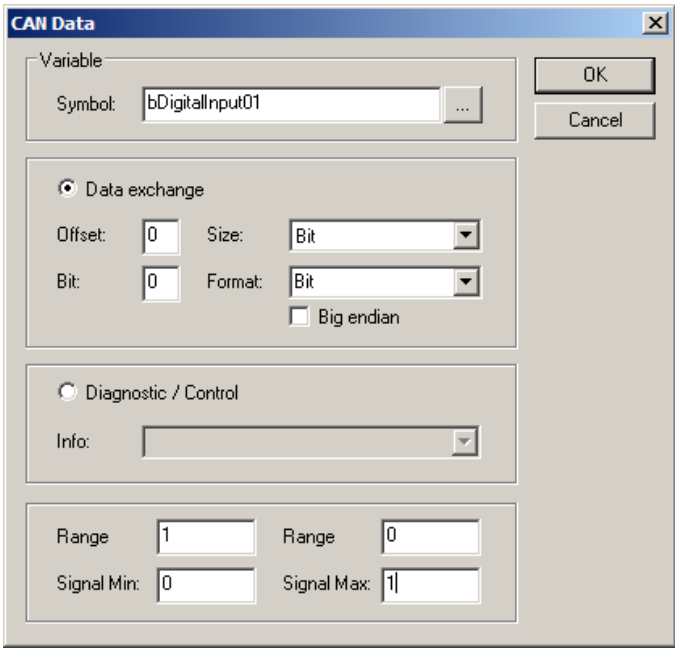
Format : Puisqu'il s'agit du TPDO1/RPDO1, spécifier un bit.

Big endian : Inversion des octets MSB et LSB dans le cas d'un mot. Non requis pour les systèmes CANopen.

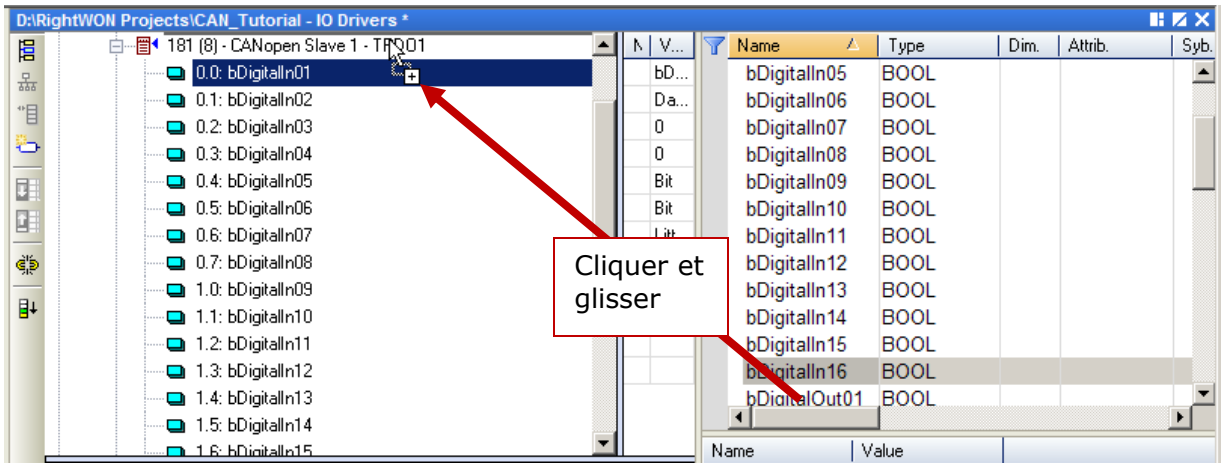
Diagnostic/Control : Cocher pour déclarer des variables relatives au contrôle des échanges (non requis pour l'instant).

Range : Définis la plage d'opération de la variable. Normalement non utilisé pour les entrées/sorties numériques à moins de faire une inversion de donnée.

Signal Min/Max : Définis la plage d'opération de l'entrée/sortie. Normalement non utilisé pour les entrées/sorties numériques à moins de faire une inversion de données. L'exemple suivant illustre l'inversion d'un bit de sortie.



4- Dans le cas où les variables sont déjà créées, vous pouvez faire l'assignation à partir de la zone des variables vers le PDO.



5- Pour les besoins du tutoriel, configurer 16 entrées digitales *bDigitalIn01* à *bDigitalIn16* et les assigner aux 16 premières entrées du TPD01. De plus, configurer 16 sorties digitales *bDigitalOut01* à *bDigitalOut16* et les assigner aux 16 premières sorties du RPD01

4.8.2. Associer les variables aux entrées/sorties analogiques

Les entrées analogiques sont associées aux PDO de la manière suivante :

TPDO2 : PDO associé aux entrées analogiques 1 à 4 du coupleur CANopen.

TPDO3 : PDO associé aux entrées analogiques 5 à 8 du coupleur CANopen.

TPDO4 : PDO associé aux entrées analogiques 9 à 12 du coupleur CANopen.

RPDO2 : PDO associé aux sorties analogiques 1 à 4 du coupleur CANopen.

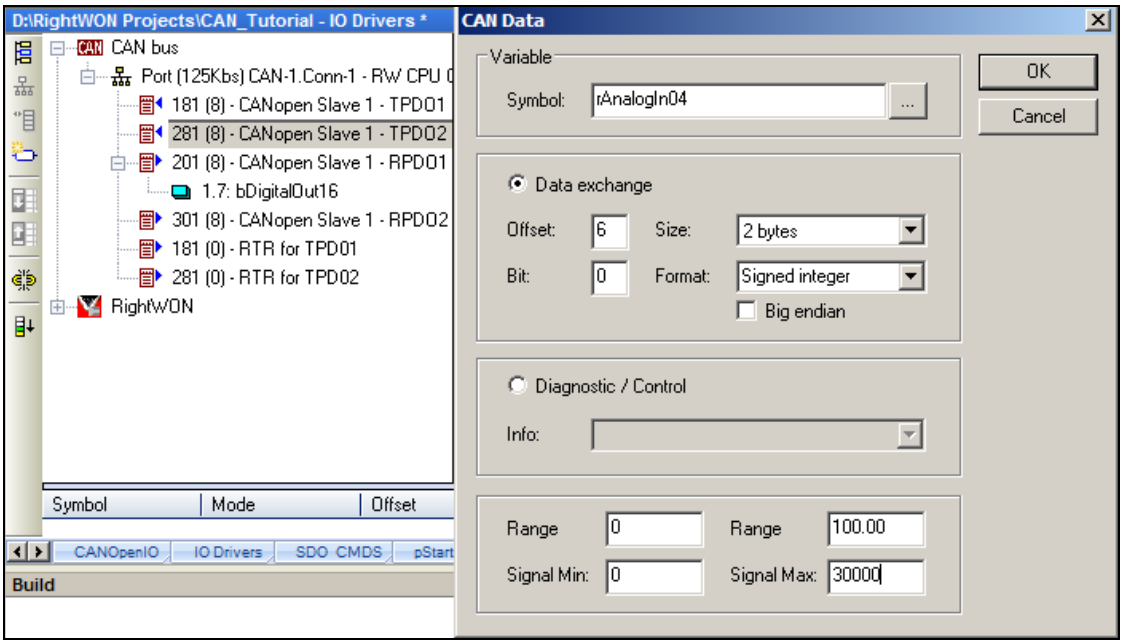
RPDO3 : PDO associé aux sorties analogiques 5 à 8 du coupleur CANopen.

RPDO4 : PDO associé aux sorties analogiques 9 à 12 du coupleur CANopen.

Pour insérer une variable analogique dans un PDO, suivre les étapes suivantes :

1- Dans la zone **Drivers E/S,** cliquer avec le bouton de droite de la souris sur le PDO désiré et sélectionner **Insérer une variable...**  .

2- La fenêtre de configuration des données du PDO s'ouvre. Cette fenêtre permet d'assigner jusqu'à 4 variables de 16 bits au PDO en utilisant un système d'offset qui désigne l'octet 0 à 7 du PDO. Pour plus de détails, voir la section « Fonctionnement des PDO associés aux entrées/sorties analogiques ».



3- Assigner les champs suivants:

Symbole: Nom de la variable associée à l'entrée/sortie. Le bouton ... permet d'insérer une variable.

Data exchange : Cocher pour l'échange des données.

Offset : Spécifier l'offset d'octet dans le PDO (0, 2, 4 ou 6 pour les données 16 bits).

Bit : Non utilisé, doit être laissé à 0.

Size : Sélectionner 1 ou 2 octets selon qu'il s'agit des données de 8 bits ou 16 bits. Dans le cas de convertisseurs 12 bits, choisir quand même 2 octets.

Format : Choisir en fonction du modèle d'entrée/sortie utilisé. Habituellement, les entrées/sorties unipolaires 0-10V ou 4-20mA sont des entiers signés.

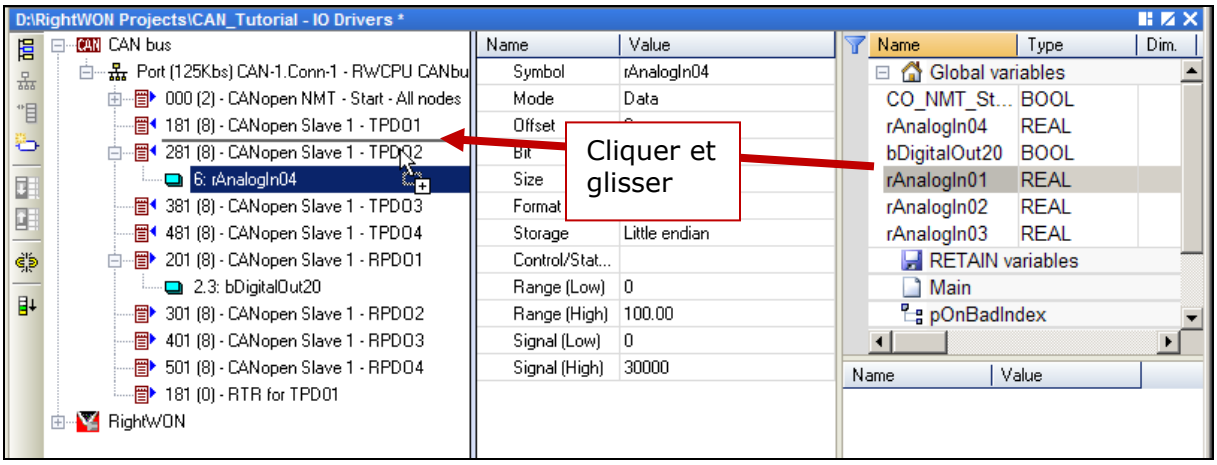
Big endian : Inversion des octets MSB et LSB dans le données. Non requis pour les systèmes CANopen.

Diagnostic/Control : Cocher pour déclarer des variables relatives au contrôle des échanges (non requis pour l'instant).

Range : Définis la plage d'opération de la variable en valeur d'ingénierie en relation avec le module.

Signal Min/Max : Définis la plage d'opération de l'entrée/sortie. L'exemple illustre une entrée analogique provenant d'un module Phoenix Contact IB IL AI 2/SF. Le module fait une conversion 4-20mA à un compte brut variant entre 0 @ 30000 (Signal Min et Max). La variable associée à l'entrée variera entre 0 et 100.00 (Range).

4- Dans le cas où les variables sont déjà créées, vous pouvez faire l'assignation à partir de la zone des variables vers le PDO.



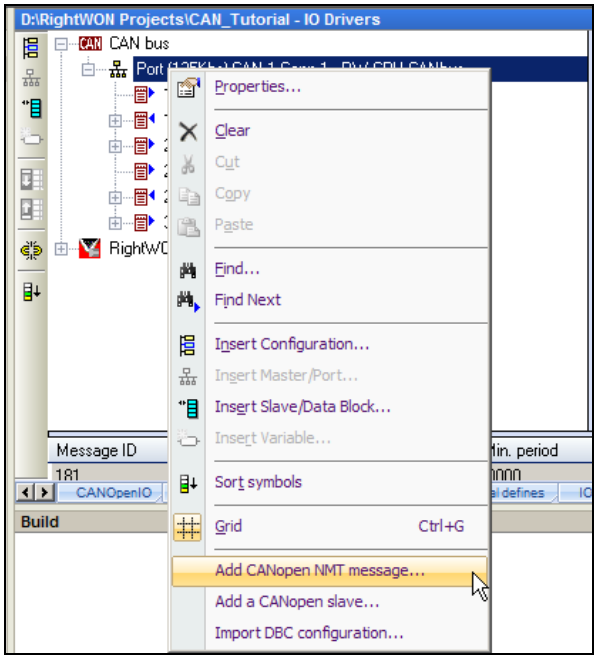
5- Pour les besoins du tutoriel, configurer 4 entrées analogiques rAnalogIn0*1* à rAnalogIn0*4* et les assigner aux 4 premières entrées du TPD02. De plus, configurer 4 entrées sorties rAnalogOut0*1* à rAnalogOut0*4* et les assigner aux 4 premières sorties du RPD02.

4.9. Créer un message NMT CANopen

Les messages NMT CANopen permettent à un programme d'envoyer des commandes pour changer l'état des nœuds CANopen. Un message NMT START NODE est requis pour mettre le coupleur CANopen en ligne.

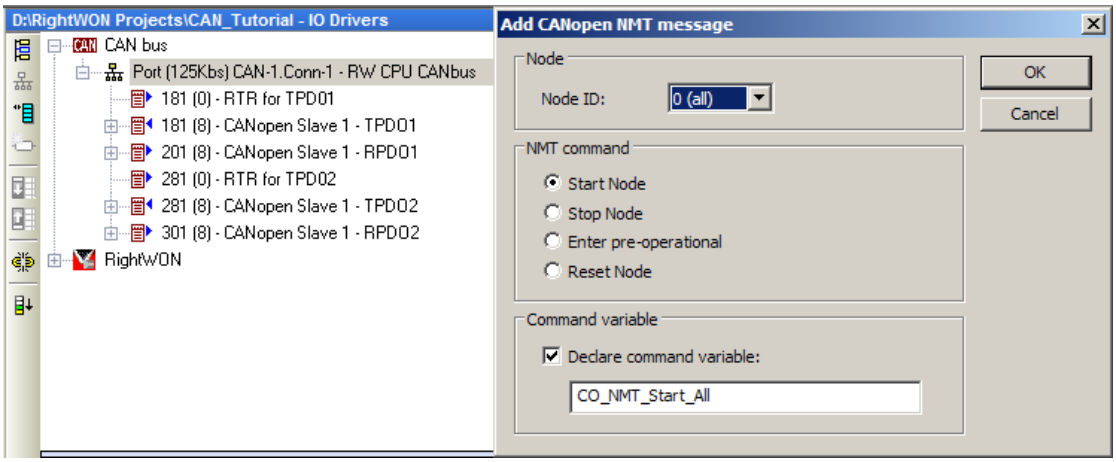
Exécuter les étapes suivantes pour insérer un message CANopen.

1- Dans la zone **Drivers E/S**, cliquer avec le bouton de droite de la souris sur **Port CAN-1.Conn-1** et sélectionner **Ajouter un message NMT CANopen...**



2- Une fenêtre de configuration des messages NMT s'ouvre. Voici la liste des paramètres :

- Dans la section **Nœud** : choisir le numéro d'identification du nœud destinataire (COB-ID).
  Note : 0 (all) permet d'envoyer le message à tous les nœuds.
- Dans la section **Commande NMT**, cliquer sur l'action désirée. Dans le cas présent, choisir **Démarrer**.
- Dans la section **Variable de Commande**, inscrire le nom de la variable qui déclenche l'envoi de la commande. Cette variable sera utilisée dans le programme pour déclencher l'envoi du message.



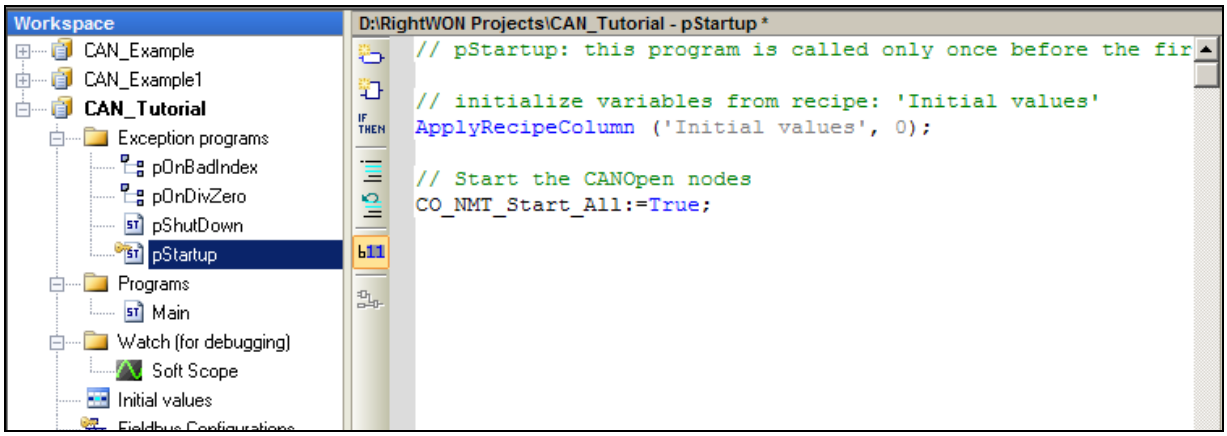
3- Cliquer sur **OK** pour insérer le message.

Après avoir inséré un message NMT CANopen, il faut déclencher l'envoi de la commande à partir d'un programme en manipulant la variable associée à la commande.

Après une mise sous tension d'un coupleur CAN, il faut démarrer le nœud.

4.10. Démarrage des nœuds avec un message NMT CANopen

Afin que le CANbus entre dans l'état opérationnel, il faut démarrer les nœuds. Pour ce faire, il faut créer un message NMT de démarrage des nœuds et déclencher l'envoi du message dans le programme d'exception de démarrage (pStartup). Pour ce faire, il suffit de lever la requête *CO_NMT_Start_All* dans le programme de pStartup appelé une seule fois au démarrage du RightWON. Le pilote en fait la remise à 0 automatiquement à l'envoi de la requête.



4.11. Mise en ligne du projet

Exécuter les étapes 1 à 10 du tutoriel de configuration du CANbus. Ce tutoriel permet de faire l'échange d'entrées/sorties analogiques et digitales entre le RightWON et le coupleur CANopen.

Le tutoriel est réalisé en fonction d'un système comportant 16 entrées/sorties digitales et 4 entrées/sorties analogiques. Vous pouvez le modifier en fonction de vos besoins en supprimant les PDO non utilisés et/ou les variables non utilisées.

4.11.1. Préparation du coupleur CANopen

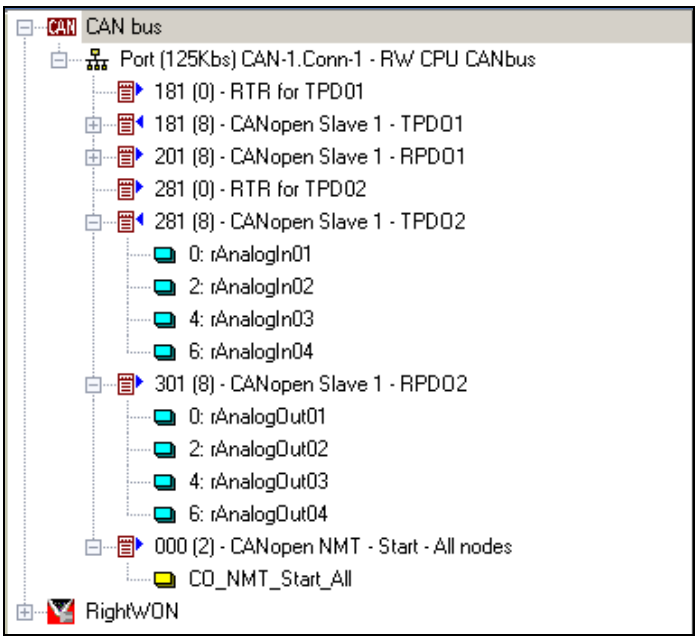
Relier votre coupleur au RightWON par l'entremise d'un câble blindé et torsadé. L'assignation des broches du connecteur CANbus du RightWON localisé en dessous de l'unité est la suivante (se référer à la spécification du module CPU - RWU010000-SP-fr):

Pin number	Assignment
1 (front)	Protective ground. Connect to cable shield
2	Not connected
3	CAN GND (reference GND)
4	CAN L
5	CAN H
6 (rear)	120-ohm termination. Jump to pin 5 to insert the termination.

Selon l'exemple du tutoriel, configurer le coupleur à un débit de 125Kbps et son adresse à 1.

4.11.2. Préparation de l'application du RightWON

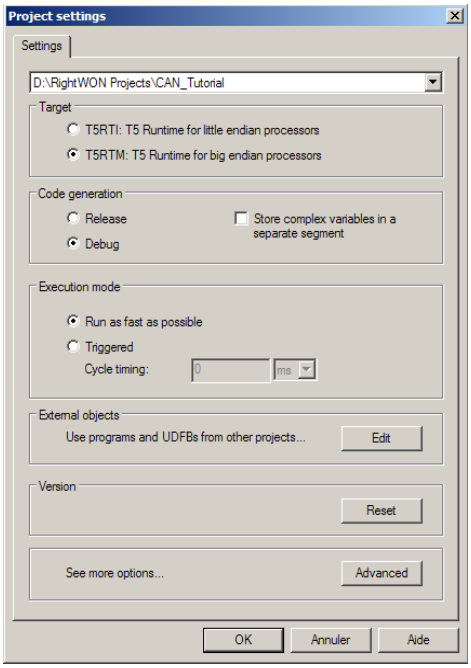
Avant de compiler et mettre le projet en ligne, s'assurer que l'arborescence de la configuration **CANbus** du tutoriel ressemble à la figure suivante. Il est à noter que le TPDO1 et le RPDO1 devraient être couplés à 16 variables chacune.



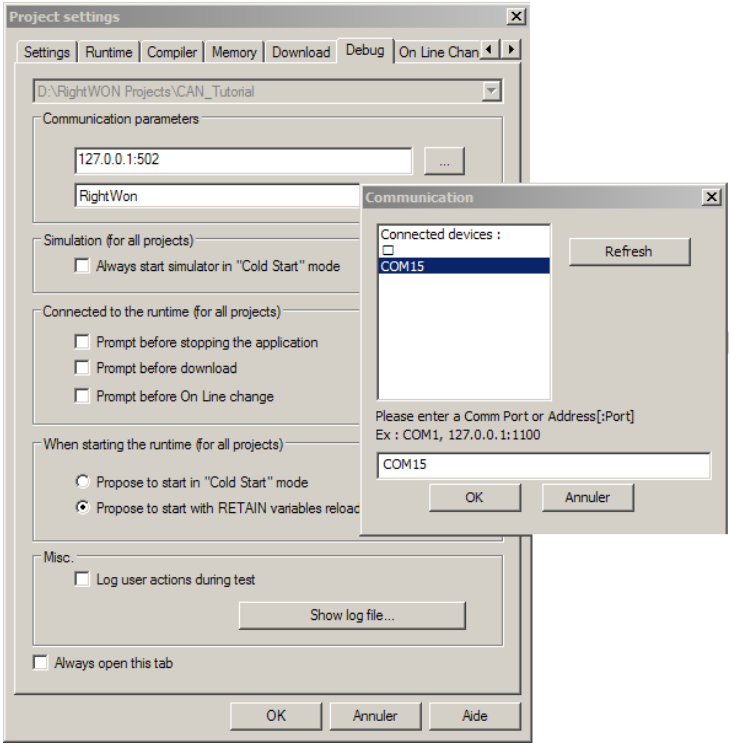
4.11.3. Préparation du projet

Avant de mettre le projet en ligne, il faut régler les options du projet :

1- Activer la commande **Projet/Settings...** et cocher la case **T5RTM : T5 for big endian processors** afin de générer le code du RightWON.

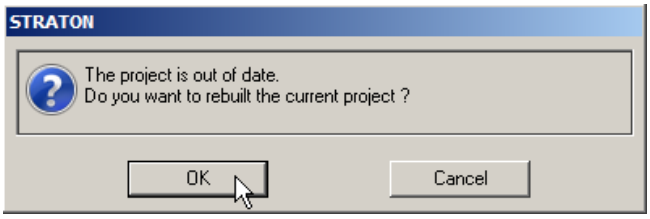


2- Activer la commande **Tools/Communication Parameters...** et activer le bouton ... pour sélectionner le port de communication utilisé pour établir la communication entre le RightWON et votre ordinateur qui exécute le RightWON Configuration Suite.

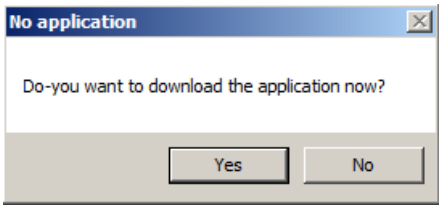


4.11.4. Mettre le projet en ligne

1- Cliquer sur l'icône **En ligne** (CTRL+F5) pour compiler et télécharger le projet. Cliquer sur **OK** afin de recompiler le projet.



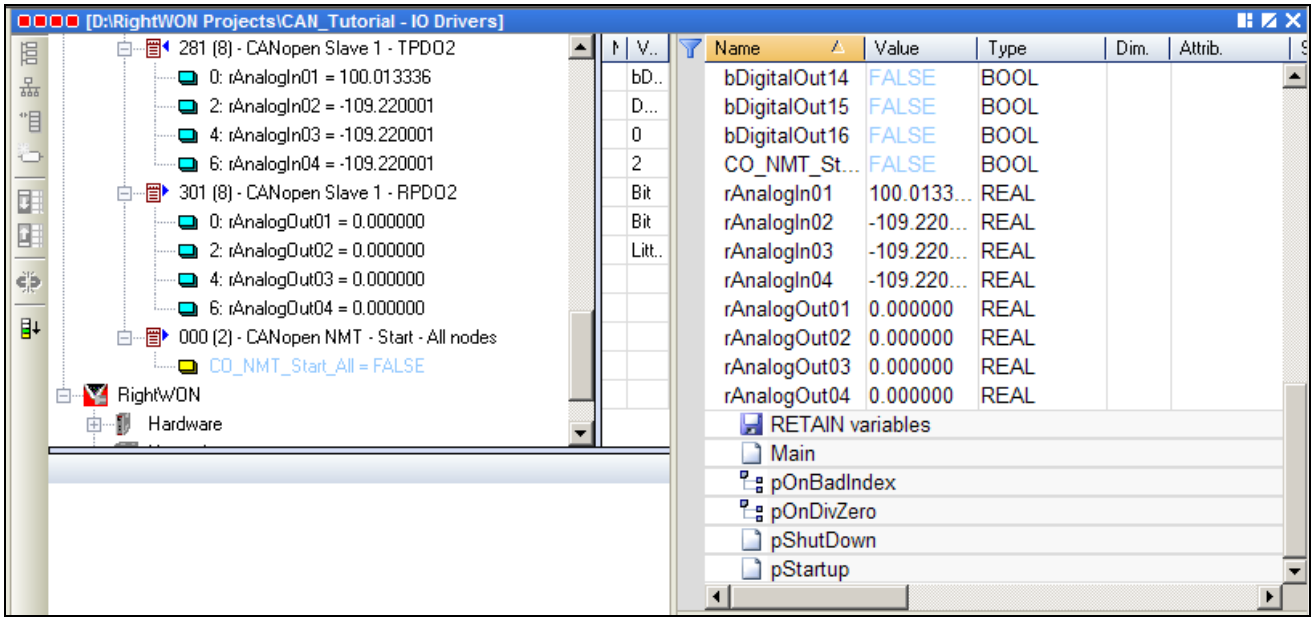
2- Cliquer sur **Oui** afin de télécharger la nouvelle version de l'application.



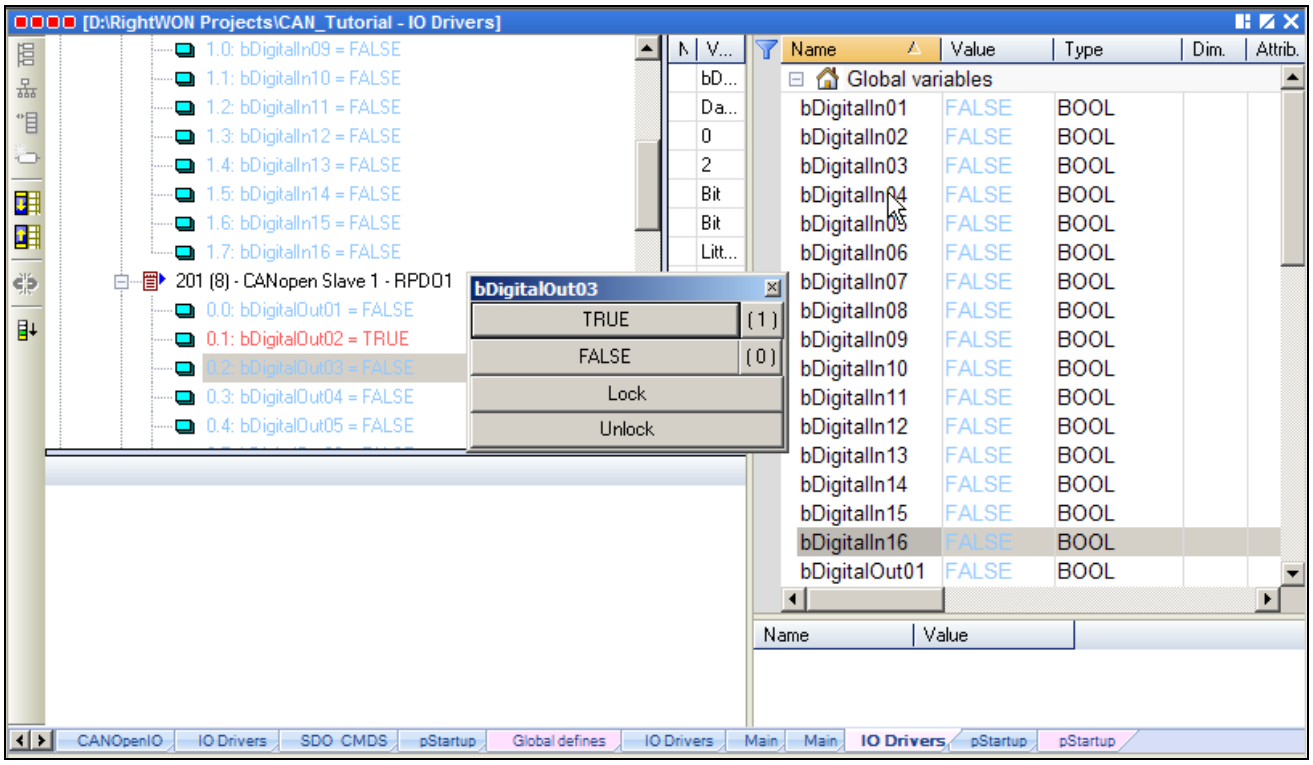
3- Le projet devrait maintenant être en mode de marche dans le système RightWON.

4.11.5. Fonctionnement de l'exemple CANbus

L'exemple CANbus permet d'envoyer et de recevoir des données entre le RightWON et le nœud 1 CANopen. La visualisation des données peut être réalisée à partir de la zone des variables ou dans l'arborescence de la configuration CANbus. Dans l'exemple suivant, une tension de 10V est injectée dans la première entrée (valeur de 100.0133 en tenant compte de la mise à l'échelle). Les 3 autres entrées analogiques affichent un nombre négatif, ce qui indique un dépassement d'échelle.

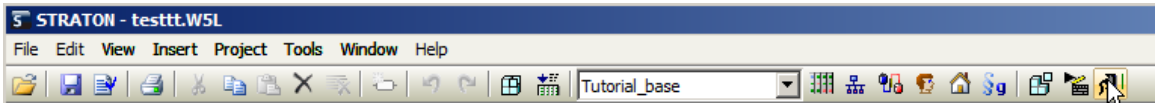


Seules les variables de sortie peuvent être forcées par l'utilisateur. Pour forcer une sortie digitale, faire un double clic sur la variable associée dans la zone Drivers E/S ou dans la zone des variables. Forcer ensuite la variable à VRAI ou FAUX. La même méthode peut être utilisée pour la mise à jour des sorties analogiques.

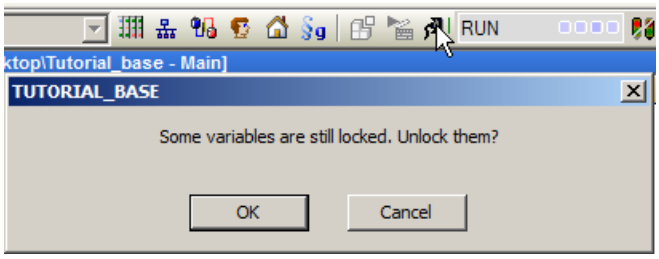


4.11.6. Sortir du mode en ligne

Pour sortir du mode en ligne, cliquer sur l'icône **En ligne**. Le programme de la cible continue à s'exécuter sans toutefois communiquer avec votre ordinateur.



Si une fenêtre s'ouvre pour vous avertir que des variables sont toujours verrouillées, cliquer sur **OK** pour les déverrouiller.



Gestion avancée du protocole CANopen

Cette section décrit la gestion avancée du protocole CANopen. Elle décrit entre autres :

- Gestion du dictionnaire (SDO), décrivant l'utilisation des SDO pour la configuration des modules d'entrée/sorties dans le dictionnaire d'objets.
- Exemple d'utilisation des requêtes SDO pour l'initialisation, décrivant un exemple de l'utilisation des requêtes SDO pour configurer les modules d'entrée/sorties et initialiser les nœuds.
- Ajout de PDO supplémentaire à la configuration CANopen, décrit comment on peut ajouter des PDO à un système pour supporter un plus grand nombre d'entrées/sorties.
- Description du projet CAN_PDO_Example, décrivant un exemple d'ajout de PDO supplémentaire à la configuration CANopen (ex. : ajout d'un TRPD05 avec l'identificateur COB-ID 182 (TPDO1 du nœud 2)).
- Utilisation des blocs de fonction CANSNDMSG et CANRCVMSG, ces fonctions permettent l'écriture et la lecture de données sur le CANbus à partir d'un programme d'application plutôt qu'à partir du configurateur de bus de terrain.

5.1. Gestion du dictionnaire (SDO)

Les services SDO (Service Data Objet) sont utilisés pour faire l'écriture ou la lecture d'objets dans le dictionnaire d'objets (OD) des modules CANopen. C'est à l'aide des SDO que peut être modifié le comportement des modules d'entrées/sorties, par exemple :

- Changer le mode de fonctionnement des entrées analogiques de façon à ce qu'elles envoient les changements lors du dépassement de la bande morte;
- Modifier les identificateurs de messages pour les systèmes qui dépassent les configurations de base;
- Paramétrer la configuration d'un module, etc.

Tout comme les RPDO et TPDO, les SDO sont identifiées par l'identificateur de message COB composé de l'identification de base plus le numéro de nœud. Ainsi, un message initié par le RightWON vers le contrôleur CANopen portera l'identificateur 604 pour le nœud 4 alors que la réponse à ce message portera le COB 584.

SDO	Identification de base du nœud 1 (hexadécimal)	Identification de base du nœud 4 (hexadécimal)
RightWON → Contrôleur CANopen	601	604
Contrôleur CANopen → RightWON	581	584

Chaque requête envoyée vers le contrôleur CANopen (601 et plus) génère une réponse de la part du coupleur. Il peut s'agir d'une donnée lue par le RightWON ou une confirmation de l'opération d'écriture.

Chaque message SDO comporte généralement 8 octets :

Command: (SDO command identifier) : Spécifie le type d'échange de données. Les commandes les plus couramment utilisées sont :

Initiate domain download : Requête d'écriture vers le coupleur CANbus:
0x23 : Transfert de 4 octets
0x2b : Transfert de 2 octets
0x2f : Transfert de 1 octet

Initiate domain upload : Requête de lecture du coupleur vers le RightWON:
0x40 : Lecture de données

Index: Index de l'objet. Par exemple, l'index 0x6006 indique quelles entrées numériques de 8 bits produisent l'envoi d'un message RPDO1 vers le RightWON (Interrupt mask any change 8-bit).

Subindex: Sous-index de l'objet. Désigne habituellement un point E/S du module. Par exemple, le sous-index 2 de l'index 0x6006 désigne le deuxième groupe de 8 bits.

Data: Désigne les données à échanger. Lors d'une requête de lecture du coupleur, ce champ est ignoré.

5.1.1. Définition des messages SDO dans le pilote CANbus

Deux méthodes peuvent être utilisées pour définir les messages SDO dans le pilote CANbus :

1. Par l'ajout d'un esclave CANopen (« Add a CANopen slave »)
2. Par l'ajout d'un message (« Insert Slave/Data block »)

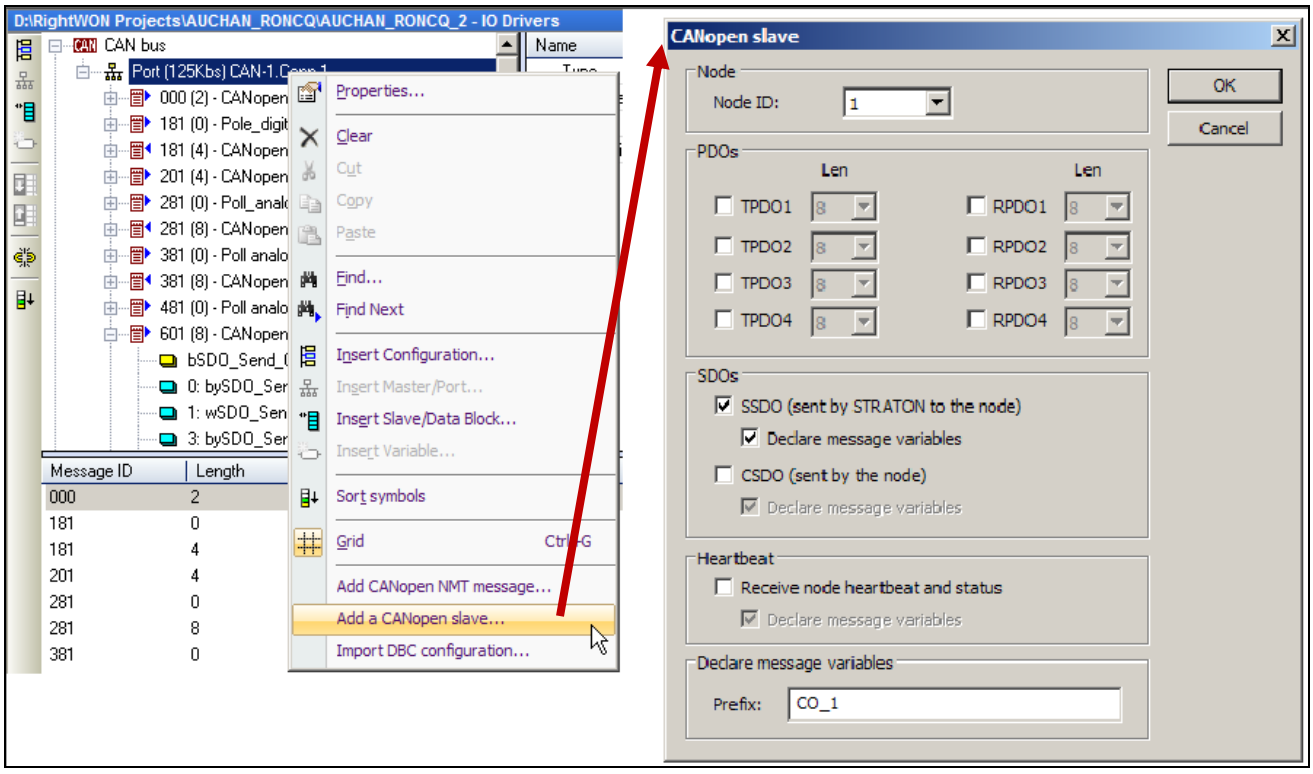
5.1.1.1. Ajout d'un esclave CANopen (« Add a CANopen slave »)

1- Accéder à la commande « Add a CANopen slave » par un clic droit sur le port CANbus.

2- Décocher tous les PDO et cocher SSDO pour les requêtes envoyées par le RightWON vers le SDO (0x601 et plus) ou cocher CSDO pour les réponses du coupleur envoyées au RightWON (0x581 et plus).

3- Cocher « Declare message variables » pour déclarer automatiquement 5 variables associées à la requête.

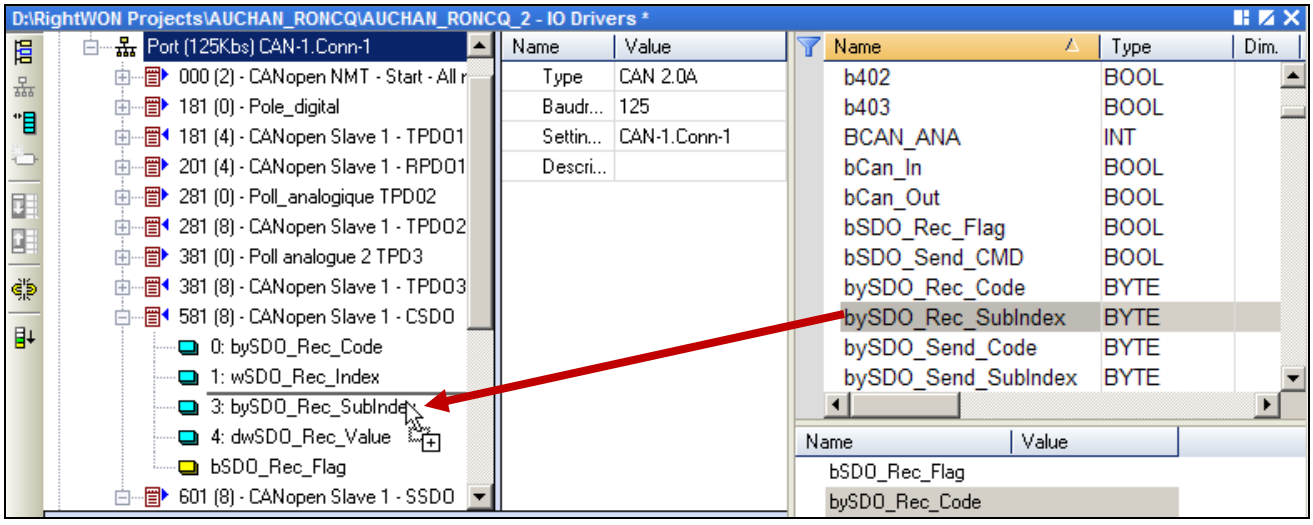
4- Identifier le préfixe aux variables dans le champ « Prefix » si vous décidez de déclarer automatiquement les variables.



5- Renommer s'il y a lieu le nom des variables. Selon les standards de programmation utilisés par Vizimax, les préfixes *b*Variable, *by*Variable, *w*Variable et *dw*Variable désignent respectivement un bit, un octet, un mot et un double mot.

6- Vous pouvez réassigner les variables aux requêtes en glissant et déposant les variables sélectionnées à partir de la zone des variables dans les requêtes.

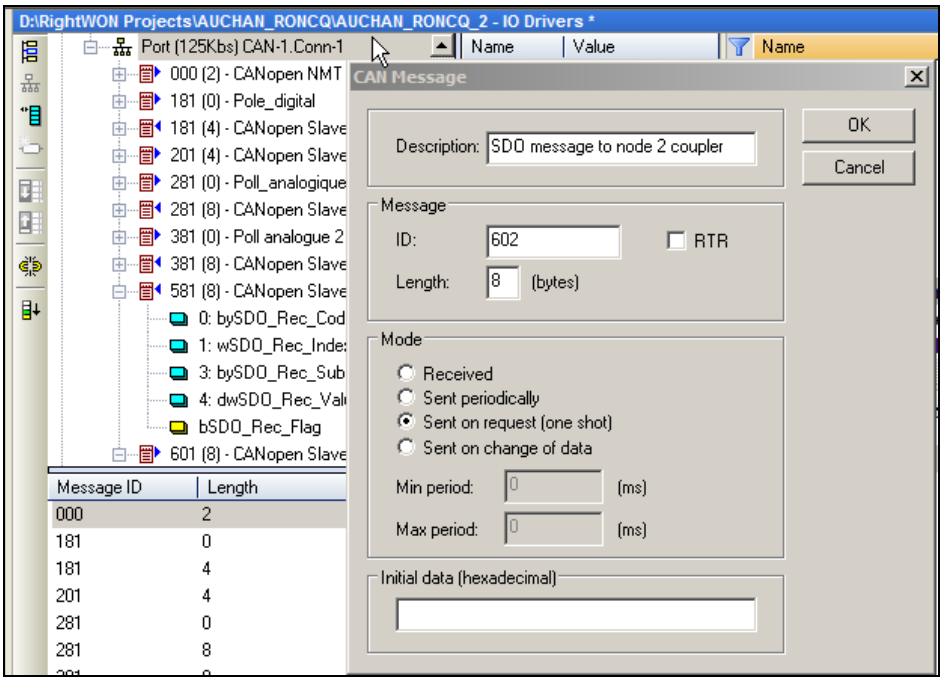
7- Répéter les étapes 1 à 6 pour générer la requête de réponse CSDO.



5.1.1.2. Ajout d'un message (« Insert Slave/Data block »)

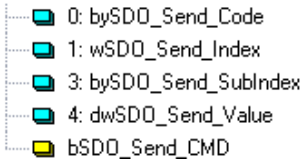
1- Sélectionner le port CANbus puis cliquer sur « Insert Slave/Data block » (outil).

2- Entrer la description du message, l'identificateur COB du message (600 + adresse du nœud), la longueur du message (8 octets) et déclarer l'envoi sur demande.

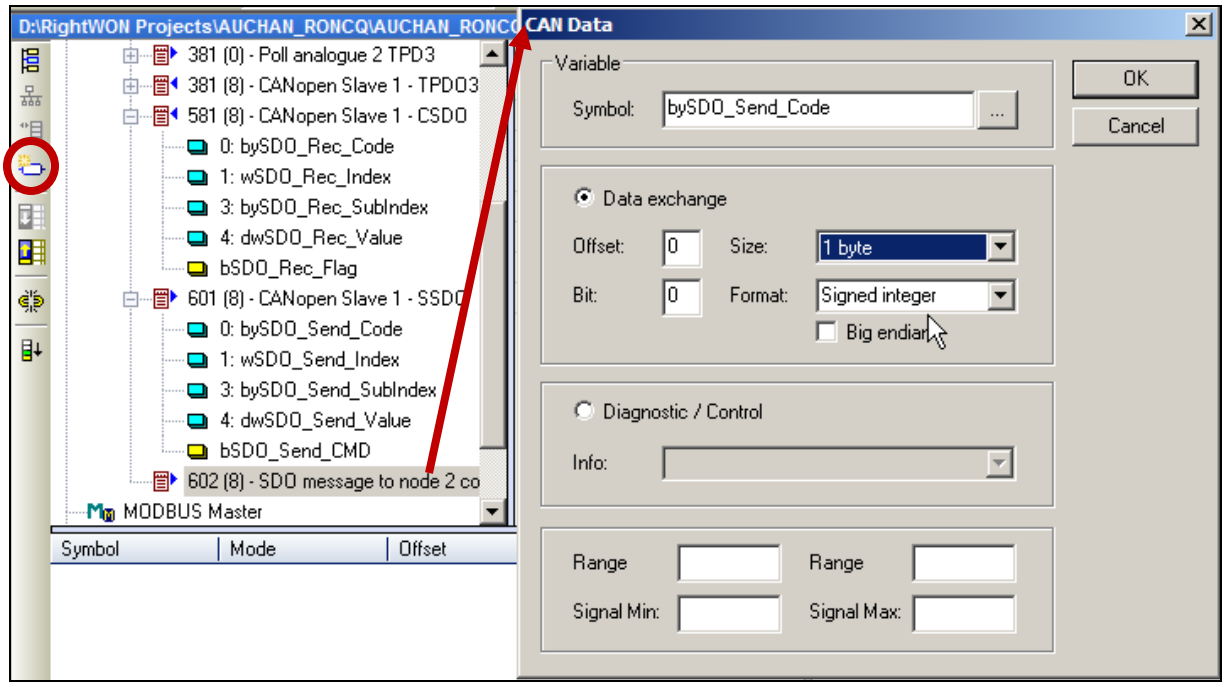


3- Dans la zone des variables, définir les 5 variables utilisées dans l'échange :

- La commande
- L'index
- Le sous-index
- Les données à échanger
- Le bit d'envoi de la requête SDO



4- Associer une à une les variables associées à la requête à l'aide de la commande « Add variable » . Alternativement, vous pouvez assigner les variables aux requêtes en glissant et déposant les variables sélectionnées à partir de la zone des variables dans les requêtes.



5- Répéter les étapes 1 à 4 pour définir la réponse (l'identificateur COB du message (580 + adresse du nœud).

5.2. Exemple d'utilisation des requêtes SDO pour l'initialisation

L'exemple « Init_Example » suivant illustre la configuration de 4 modules de 2 entrées analogiques Phoenix Contact IB IL AI 2/SF. Ces modules sont par défaut configurés en 0-10V et sont réassignés en entrées 4-20mA.

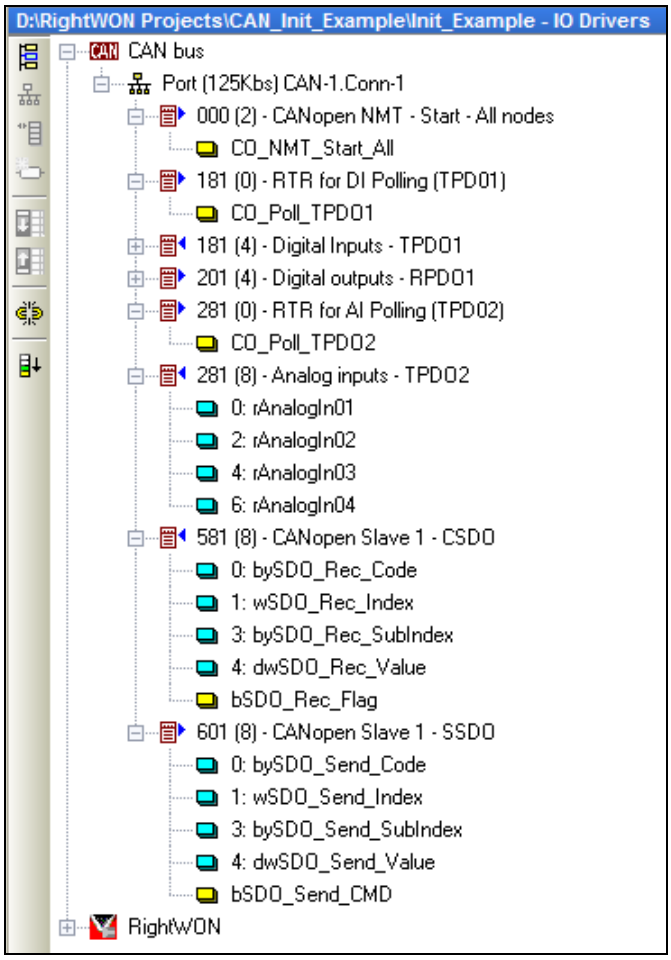
Selon la documentation du module et du contrôleur CANopen, les actions suivantes doivent être réalisées :

> Écriture de la donnée 0x800A (commande d'un mot = 0x2B) à l'index 0x2400 (AIP range) pour les 8 sous-index variant entre 1 et 8 (8 premières entrées analogiques).

Cette commande est réalisée avant le démarrage des nœuds.

5.2.1. Configuration du CANopen

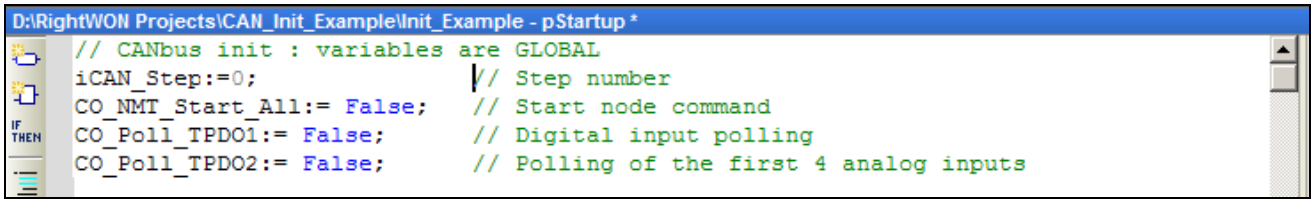
Dans cet exemple, les entrées analogiques et numériques sont rapatriées par une requête RTR sur demande (contrôlée par programmation). Les SDO sont gérées par les blocs 581 et 601, voir définition des messages SDO dans le pilote CANbus.



5.2.2. Programme de démarrage (pStartup)

Le programme de démarrage fait l'initialisation de quelques variables nécessaires à la gestion du coupleur CANopen.

Dans cet exemple, les variables CO_Poll_TPDO1 et CO_Poll_TPDO2 servent à rapatrier, par une requête RTR sur demande, les entrées analogiques et numériques.



5.2.3. Programme de gestion du CANopen

Le programme CANopen_Init exécute dans les étapes (iCAN_Step) :

1- Étape 0, 2, 4, 5, 7, 10, 12, 14 : envoie successivement les requêtes d'écriture SDO au contrôleur en spécifiant à chaque fois :
   - la commande (**Command**)
   - l'**Index**

- le sous-index (**Subindex**). Dans cet exemple, le sous-index varie de 1 à 8 pour chaque commande SDO.
- la donnée (**Data**)

Dans cet exemple, la commande, l'index et la valeur ont été définis en initialisation au début du programme, mais pourraient être définis dans chaque étape.

2- Étape 1, 3,5, 7,9, 11, 13, 15 : attend la réponse du contrôleur à chaque requête. Il est à noter que la réponse n'est pas validée.

3- Étape 16 : par la suite, les nœuds sont démarrés par une requête NMT.

4- Étape 17 : les entrées analogiques et numériques sont rapatriées sur un front montant (fonction r_trig) de temporisateurs de type Blink qui envoie sur demande une requête RTR.

```
// Initialize the data to be sent on the CANOpen controller
bSDO_Send_CMD:=0;
bySDO_Send_Code:=16#2b;     // Command : Write 2 bytes
wSDO_Send_Index:=16#2400;   // Data : Phoenix modules AIP Range
dwSDO_Send_Value:=16#800A; // Configuration data, 16-bit mean, IB IL, 4-20mA

CASE iCAN_Step OF
    //Initialize the first 8 analog inputs
    0,2,4,6,8,10,12,14:
        // The sub-index designate the channel number.
        // Index 0 is the number of analog inputs
        bySDO_Send_SubIndex:= ANY_TO_USINT(1 + (iCAN_Step/2));
        // Go to next step
        iCAN_Step := iCAN_Step + 1;
        // Send the SDO command
        bSDO_Send_CMD:=True;
    // Wait for the SDO response
    1,3,5,7,9,11,13,15:
        // The send command is valid for one cycle
        bSDO_Send_CMD:=False;
        // Flag shoud be on when receiving the answer
        if bSDO_Rec_Flag = True then
           iCAN_Step := iCAN_Step + 1; // Go to next step
        end_if;
    16:
    // Start the nodes
       CO_NMT_Start_All := True;    // Noeuds CANOpen initialisés
       iCAN_Step := iCAN_Step + 1; // Go to next step
    17:
    // Periodic scanning of the inputs
       // Timer for DIs
       T_Blink_DI(True, T#60000ms); // See Blink timer in Help
       T_Blink_AI(True, T#1000ms);
       // Detect rising edge of timers
       r_trig_DI(T_Blink_DI.Q); // See r_trig function in Help
       r_trig_AI(T_Blink_AI.Q);
       // Each Poll request must have an independant variable
       CO_Poll_TPDO1 := r_trig_DI.Q;
       CO_Poll_TPDO2 := r_trig_AI.Q;
     ELSE
       iCAN_Step := 17;
END_CASE;
```

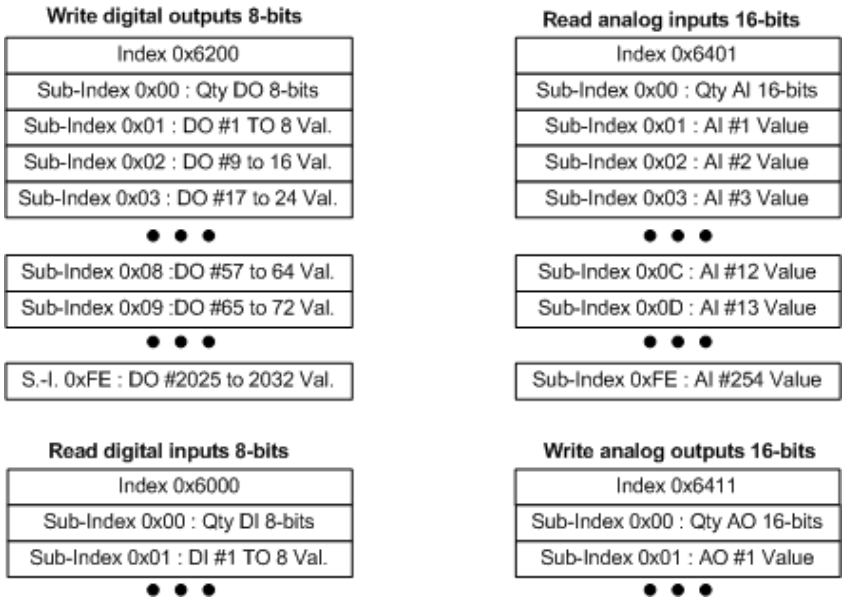
Note : L'envoi de plusieurs requêtes est facilité par l'utilisation de recettes de commande, voir la description du projet CAN_PDO_Example.

5.3. Ajout de PDO supplémentaire à la configuration CANopen

Comme préalablement mentionnés, les coupleurs CANopen conformes à la spécification DS401 permettent de supporter sans aucune configuration jusqu'à 12 entrées analogiques, 12 sorties analogiques, 64 entrées numériques et 64 sorties numériques. L'exemple « CAN_PDO_Example » illustre comment on peut ajouter des PDO à un système pour supporter un plus grand nombre d'entrées/sorties. Avant de passer aux explications du projet CAN_PDO_Example, les principes de base d'utilisation des registres associés aux PDO sont décrits dans les sous-sections suivantes.

5.3.1. Commandes SDO de lecture/écriture des entrées/sorties

Il est possible au RightWON de faire la lecture des entrées ou la mise à jour des sorties à l'aide des commandes/réponses SDO. Selon la figure suivante, une requête de lecture (0x40) à l'index 0x6401 et sous-index 00 retourne un octet indiquant le nombre d'entrées analogiques de 16 bits reliées au coupleur CANopen. La lecture de l'index 0x6401 et sous index 12 (0x0C) retourne la valeur de l'entrée analogique #12 sur un entier signé de 16 bits.



De la même manière, il est possible de faire la mise à jour de 8 bits sur le troisième module de sortie à l'aide d'une commande d'écriture d'un octet (0x2F) à l'index 0x6200 et sous-index 3. Cette méthode de lecture/écriture des entrées/sorties supplémentaires qui excèdent les PDO de base est supportée par le RightWON, mais génère un trafic de communication important sur le CANbus. Il est alors préférable de construire des PDO supplémentaires en utilisant les registres de mapping des PDO.

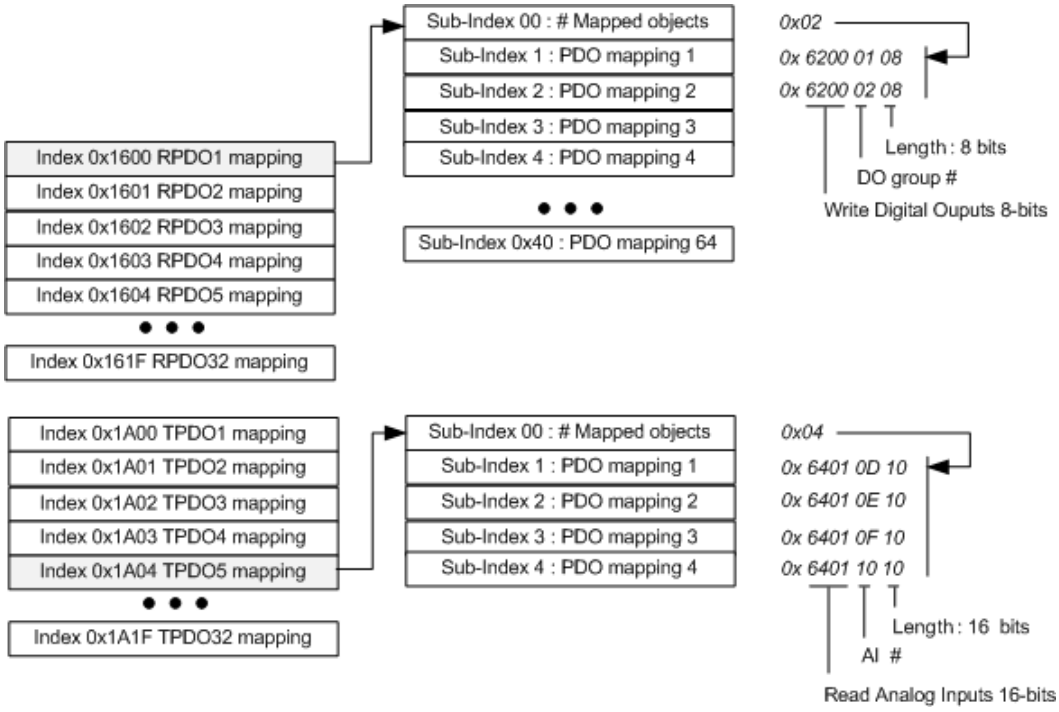
5.3.2. Registres de mapping des PDO

Les registres de mapping permettent d'assigner le contenu aux PDO. Le nombre de registres de mapping dépend du manufacturier du coupleur. Par exemple, le coupleur CANopen IL CAN BK-TC de Phoenix Contact supporte la définition d'un maximum de 32 RPDO et 32 TPDO incluant les PDO de base (4 de chaque type). Tel qu'illustré à la figure suivante, il existe un enregistrement (index) par PDO. Le sous-index 0 contient le nombre d'objets à mapper alors que les sous-index suivants définissent le mapping des objets, c'est-à-dire la liste des commandes SDO de lecture/écriture à effectuer sur les modules.

L'exemple de la figure suivante illustre le mapping du RPDO1 (index 0x1600) d'un coupleur muni de 2 modules de sorties digitales de 8-bits (sous-index 00 = 2). Les sous-index 1 et 2 contiennent alors la commande d'écriture SDO de 8 bits sur le premier et le second module. S'il y avait eu à supporter 72 sorties digitales, les 64 premières sorties seraient automatiquement assignées au RPD01. Il faudrait alors créer par exemple le RPD05 (index 0x1605) et assigner 1 objet (sous-index 00) avec un mapping de 0x6200 09 08 (9^{ième} groupe de sorties de 8 bits).

L'exemple illustre aussi le mapping du TPDO5 (index 1A04) comprenant 4 entrées analogiques (entrées #13 à #16). Le mapping contient le nombre (sous-index 0 = 4) et la liste des entrées du TPDO exprimées en commandes de lecture.

En plus de définir le mapping des PDO, il faut aussi faire l'assignation de l'identificateur COB-ID.



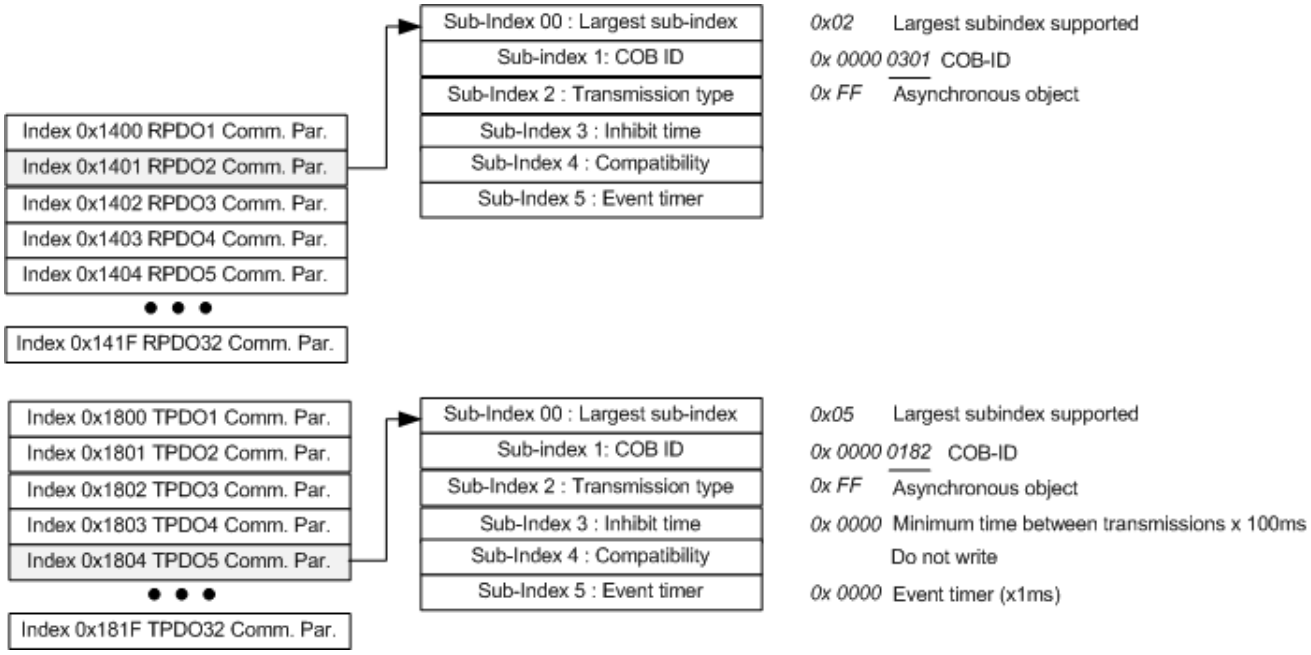
5.3.3. Assignation d'un identificateur à un PDO

L'identificateur et le comportement des RPDO et des TPDO sont gérés par les registres de paramètres de communication illustrés à la figure suivante. Un registre est associé à chaque PDO. La programmation et le nombre de ces registres varient en fonction du manufacturier du coupleur CANopen. Par exemple, le coupleur CANopen IL CAN BK-TC de Phoenix Contact supporte la définition d'un maximum de 32 RPDO et 32 TPDO incluant les PDO de base (4 de chaque type).

L'exemple de la figure suivante illustre la configuration par défaut d'un système comportant au minimum 4 sorties analogiques. Généralement, les RPDO ne supportent que 2 entrées de sous-index (sous-index 0 = 2). Le sous-index 1 est égal au COB-ID du RPDO (301 dans le cas du RPD02 ou 0x80000000 si le RPDO de l'index correspondant n'existe pas). L'index 02 est égal à 0xFF pour les RPDO asynchrones (non reliés à une commande SYNC, mis à jour directement ou par requête RTR).

L'exemple illustre aussi la configuration du TPD05. L'identificateur 182 (TPDO1 du nœud 2) a été choisi pour le TPDO5 puisque le protocole CANopen de base ne supporte que 8 identificateurs de PDO par nœud (se référer à la section adressage des PDO et assignation de PDO supplémentaire à un nœud). Une requête RTR au TPDO1 du nœud 2 force le coupleur du nœud 1 à envoyer son TPDO5.

Il est à noter qu'une séquence stricte de définition et d'assignation des PDO doit être respectée.



5.3.4. Séquence de définition et d'assignation d'un PDO

La création ou la modification d'un mapping de PDO et son assignation à un identificateur doit se faire selon la procédure suivante, se référer à composition d'un message SDO :

Note : Se référer au manuel de votre coupleur CANopen pour les index, les sous-index et les commandes (double-mot) à réaliser sur le module PDO pour mapper les entrées ou les sorties analogiques ou digitales.

1- Choisir l'**Index** du PDO dans le registre des paramètres de communication en fonction du PDO à créer (voir la section « Assignation d'un identificateur à un PDO »). Invalider l'assignation de l'identificateur en écrivant 0x8000 0000 (**Data**) au sous-index (**Subindex**) du COB-ID.

2- Choisir l'**Index** du PDO dans le registre des paramètres de mapping en fonction du PDO à créer (voir la section « Registres de mapping des PDO »). Détruire le mapping existant en écrivant 0 (**Data**) dans le sous-index (**Subindex**) du nombre d'objets mappés dans le PDO.

3- Écrire la commande (double-mot) (**Data**) à réaliser sur le module PDO pour mapper les entrées ou les sorties analogiques ou digitales (sous-index (**Subindex**) du n-objet à mapper). L'**Index** du PDO est dans le registre des paramètres de mapping en fonction du PDO à créer (même index que l'étape 2).

4- Écrire le nombre de mapping *n* réalisé à l'étape précédente (**Data**) au sous-index (**Subindex**) du nombre d'objets mappés dans le PDO. Choisir l'**Index** du PDO dans le registre des paramètres de mapping en fonction du PDO à créer (même index et sous index que l'étape 2).

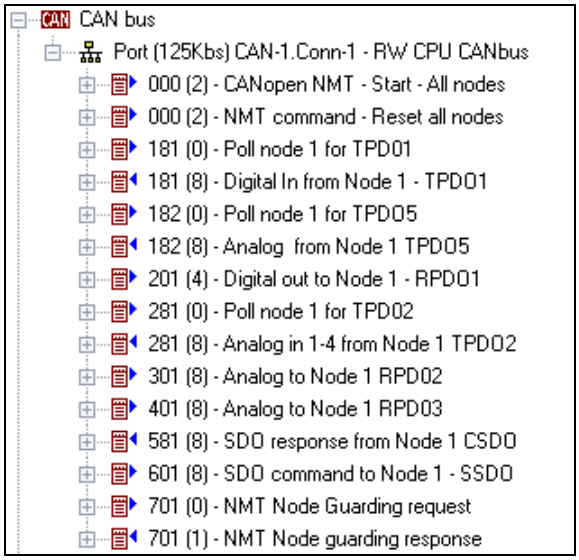
5- Attribuer un identificateur COB-ID au PDO créé. L'index du PDO est dans le registre des paramètres de communication, sous-index du COB-ID (même index et sous index que l'étape 1).

5.4. Description du projet CAN_PDO_Example

Le programme CAN_PDO_Example montre un exemple de définition du TPDO5 avec l'identificateur COB-ID de 182 (TPDO1 du nœud 2). Les 2 premières entrées analogiques du coupleur sont assignées au TPDO5 en plus du TPDO2.

5.4.1. Configuration du CANopen

Dans cet exemple, les entrées analogiques et numériques sont rapatriées par une requête RTR sur demande (contrôlée par programmation). Les SDO sont gérées par les blocs 581 et 601. De plus, le programme fait la surveillance du coupleur avec des requêtes RTR NMT Node Guarding (701).



5.4.2. Définitions

Les symboles suivants sont utilisés dans le programme. Ces définitions sont globales.

```
// CANbus defines

#define WriteByte BYTE16#2F  // SDO command identifier for initiate domain download  e,s=1, n=3
#define WriteWord BYTE16#2B  // SDO command identifier for initiate domain download  e,s=1, n=2
#define WriteDWord BYTE16#23 // SDO command identifier for initiate domain download  e,s=1, n=0
#define ReadSDO BYTE16#40    // SDO command identifier for initiate domain upload
#define Preoperational 16#7F // CAN coupler is preoperational
#define Operational 16#05    // CAN coupler is operational
#define NodeGuardAnswerMask 16#7F // Node Guard Answer mask
#define SDOAnswerMask 16#E0     // SDO command answer mask
#define SDODownloadAnswer 16#60 // SDO Initiate Domain Download valid answer
#define CANInit 0               // CANbus initialization in progress
#define CANInitError -1         // CANbus initialization error
#define CANRunning 1            // CANbus is now running
#define CAN_NoHBAnswer -2       // CANbus no answer to heart beat request
#define CAN_BadNode  -3         // CANbus bad answer to heart beat
```

5.4.3. Programme de démarrage (pStartup)

Le programme de démarrage fait l'initialisation de la variable iCAN_Step nécessaires à la gestion du coupleur CANopen.

```
D:\RightWON Projects\CAN_PDO_Example - pStartup
// pStartup: this program is executed once
// and continues with the other programs
// in the same cycle

// CANbus variables initialization
iCAN_Step:=0;    // Step number of the CANbus initialization
```

5.4.4. Recette de commandes

Les commandes SDO sont programmées dans la recette SDO_CMDS de façon à pouvoir être manipulées en séquence (colonne 1 et plus). Se référer à séquence de définition et d'assignation d'un PDO. La recette contient tous les champs de la commande SDO :

Nom(Name) : Contiens les champs de la commande SDO.

bySDO_Send_Code (**Command**): Identificateur de commande SDO.
wSDO_Send_Index (**Index**): Index du dictionnaire d'objets.
bySDO_Send_SubIndex (**Subindex**): Sous-index du dictionnaire d'objets.
dwSDO_Send_Value (***Data***): Donnée à écrire dans le dictionnaire.

(Voir **Command**, **Index**, **Subindex**, **Data** dans la section « Gestion du dictionnaire SDO »)

Valeur (Value) : Lorsque l'application est En Ligne, les valeurs des variables sont affichées dans cette colonne. Placez le curseur sur la valeur de la variable pour ouvrir une info-bulle où la valeur est affichée en hexadécimal.

0- Le champ *wSDO_Send_Index* associé à la colonne (**0**) contient le nombre de commandes SDO contenues dans la recette. Dans cet exemple, il y a 8 commandes SDO (colonne 1 à 8), inscrire dans *wSDO_Send_Index* de la colonne **0** : *WORD#8* (ex. : s'il y aurait 32 SDO (colonne 1 à 32), inscrire *WORD#32*).

SDO_CMDS.rcp

Name	Value	0	1	2	3
bySDO_Send_Code		BYTE#0	BYTE#16#23	BYTE#16#2F	BYTE#16#23
wSDO_Send_Index		WORD#8	Word#16#1804	WORD#16#1A04	WORD#16#1A04
bySDO_Send_SubIndex		BYTE#0	BYTE#16#01	BYTE#16#0	BYTE#16#01
dwSDO_Send_Value		DWORD#0	DWORD#16#80000000	DWORD#16#0	DWORD#16#64010110

SDO_CMDS.rcp

	4	5	6	7	8
	BYTE#16#23	BYTE#16#2F	BYTE#16#23	BYTE#16#2B	BYTE#16#2B
4	WORD#16#1A04	WORD#16#1A04	Word#16#1804	Word#16#2400	Word#16#2400
	BYTE#16#02	BYTE#16#0	BYTE#16#01	BYTE#16#01	BYTE#16#02
010110	DWORD#16#64010210	DWORD#16#02	DWORD#16#00000182	DWORD#16#800A	DWORD#16#800A

Les colonnes numérotées 1 à 8 sont les commandes SDO et réalisent les fonctions suivantes (Se référer à la section Séquence de définition et d'assignation d'un PDO.) :

NOTE : Pour cet exemple, on se réfère au manuel de l'utilisateur du coupleur CANopen IL CAN BK-TC de Phoenix Contact. Les commandes SDO 1 à 6 définissent un TPDO5 dans le nœud 1 avec l'identificateur du TPDO1 du nœud 2 (COB-ID 182). Les commandes SDO 7 et 8 assignent les 2 premières entrées analogiques du coupleur au TPDO5 en plus du TPDO2.

SDO_CMDS.rcp

Name	Value	0	1	2	3
bySDO_Send_Code		BYTE#0	BYTE#16#23	BYTE#16#2F	BYTE#16#23
wSDO_Send_Index		WORD#8	Word#16#1804	WORD#16#1A04	WORD#16#1A04
bySDO_Send_SubIndex		BYTE#0	BYTE#16#01	BYTE#16#0	BYTE#16#01
dwSDO_Send_Value		DWORD#0	DWORD#16#80000000	DWORD#16#0	DWORD#16#64010110

1- Note : Se référer au manuel de votre coupleur CANopen.

Choisir l'index du PDO dans le registre des paramètres de communication en fonction du PDO à créer (ex. : TPDO5 : ***Send_Index*** #16#*1804*).

Invalider l'assignation de l'identificateur en écrivant 0x8000 0000 (***Send_Value*** *#16#80000000*) au sous-index du COB-ID (ex. : ***Send_Subindex*** *#1*).

Dans ***Send_Code***, inscrire *16#23* (transfert de 4 octets de données (*Send_Code*)).

Index (hex)	Object	Name	Data type	Access	M/O
Transmit PDO Communication Parameter					
1800	RECORD	1st transmit PDO Par.	PDO CommPar (20_{hex})	RW	M/O
1801	RECORD	2nd transmit PDO Par.	PDO CommPar (20_{hex})	RW	M/O
...	...	...	...	...	...
181F	RECORD	32nd transmit PDO Par.	PDO CommPar (20_{hex})	RW	M/O

B 7.25 Object 1800_{hex} – $181F_{hex}$: Transmit PDO Communication Parameter

Contains the mapping for the PDOs the device is able to transmit.

a. Object description

Index	1800_{hex} — $181F_{hex}$
Name	Transmit PDO Communication Parameter
Object	RECORD
Data type	PDO CommPar
Category	Conditional; mandatory for each supported PDO

b. Entry description

Subindex	0_{hex}
Description	Largest subindex supported

Subindex	1_{hex}
Description	COB-ID used by PDO
Entry category	Mandatory
Access	RO; RW if COB-ID can be configured
PDO mapping	No
Value range	UNSIGNED32 (Figure 65)
Default value	Index 1800_{hex}: 180_{hex} + Node-ID, Index 1801_{hex}: 280_{hex} + Node-ID, Index 1802_{hex}: 380_{hex} + Node-ID, Index 1803_{hex}: 480_{hex} + Node-ID, Index 1804_{hex} - $18FF_{hex}$: Disabled

Subindex	2_{hex}
Description	Transmission type

Subindex	3_{hex}
Description	Inhibit time

Subindex	4_{hex}
Description	Reserved

Subindex	5_{hex}
Description	Event timer

SDO_CMDS.rcp

Name	Value	0	1	2	3
bySDO_Send_Code		BYTE#0	BYTE#16#23	BYTE#16#2F	BYTE#16#23
wSDO_Send_Index		WORD#8	Word#16#1804	WORD#16#1A04	WORD#16#1A04
bySDO_Send_SubIndex		BYTE#0	BYTE#16#01	BYTE#16#0	BYTE#16#01
dwSDO_Send_Value		DWORD#0	DWORD#16#80000000	DWORD#16#0	DWORD#16#64010110

2- Note : Se référer au manuel de votre coupleur CANopen.

Choisir l'index du PDO dans le registre des paramètres de mapping en fonction du PDO à créer (ex. : TPDO5 : ***Send_Index*** #16*#1A04*).

Détruire le mapping existant en écrivant 0 (***Send_Value*** *#16#0*) dans le sous-index du nombre d'objets mappés dans le PDO (ex. : ***Send_Subindex*** *#0*).

Dans ***Send_Code***, inscrire *16#2f* (transfert de 1 octet de données (*Send_Value*)).

Index (hex)	Object	Name		Data type	Access	M/O
Transmit PDO Mapping Parameter						
1A00	RECORD	1st transmit PDO		PDO Mapping (21_{hex})	RW	M/O
1A01	RECORD	2nd transmit PDO		PDO Mapping (21_{hex})	RW	M/O
...	...	...		...	...	...
1A1F	RECORD	32nd transmit PDO		PDO Mapping (21_{hex})	RW	M/O

B 7.26 Object $1A00_{hex}$ – $1A1F_{hex}$: Transmit PDO Mapping Parameter

Contains the mapping for the PDOs, which the device can transmit.

a. Object description

Index	$1A00_{hex}$ — $1A1F_{hex}$
Name	Transmit PDO Mapping Parameter
Object	RECORD
Data type	PDO mapping
Category	Conditional; mandatory for each supported PDO

b. Entry description

Subindex	0_{hex}
Description	Number of mapped application objects in PDO
Entry category	Mandatory
Access	RO; RW if dynamic mapping is supported
PDO mapping	No
Value range	0: Deactivated 1 - 64: Activated
Default value	(Device profile dependent)

Subindex	1_{hex} - 40_{hex}
Description	PDO mapping for the n-th application object to be mapped

SDO_CMDS.rcp

Name	0	3	4	5
bySDO_Send_Code	... ..	BYTE#16#23	BYTE#16#23	BYTE#16#2F
wSDO_Send_Index	... ..	WORD#16#1A04	WORD#16#1A04	WORD#16#1A04
bySDO_Send_SubIndex	... ..	BYTE#16#01	BYTE#16#02	BYTE#16#0
dwSDO_Send_Value	... ..	DWORD#16#64010110	DWORD#16#64010210	DWORD#16#02

Note : Se référer au manuel de votre coupleur CANopen.

3- Mapper la commande (***Send_Value*** **:** *#16#64010110*) de l'entrée analogique #1 au sous-index du n-objet à mapper (aussi mappée dans TPD01 du nœud 2).

Send_Index du TPDO5 : 16*#1A04*, ***Send_Subindex*** : *#1.* ***Send_Code***: *#16#23* (transfert de 4 octets de données (*Send*_Value)).

4- Mapper la commande (***Send_Value*** **:** *#16#64010210*) de l'entrée analogique #2 au sous-index du n-objet à mapper (aussi mappée dans TPD01 du nœud 2).

Send_Index du TPDO5 : 16*#1A04*, ***Send_Subindex*** : *#1.* ***Send_Code***: *#16#23* (transfert de 4 octets de données (*Send_Value*)).

B 7.26 Object $1A00_{hex}$ – $1A1F_{hex}$: Transmit PDO Mapping Parameter

Contains the mapping for the PDOs, which the device can transmit.

a. Object description

Index	$1A00_{hex}$ — $1A1F_{hex}$
Name	Transmit PDO Mapping Parameter
Object	RECORD
Data type	PDO mapping
Category	Conditional; mandatory for each supported PDO

b. Entry description

Subindex	0_{hex}
Description	Number of mapped application objects in PDO
Subindex	1_{hex} - 40_{hex}
Description	PDO mapping for the n-th application object to be mapped
Entry category	Conditional, depends on number and size of objects to be mapped
Access	RW
PDO mapping	No
Value range	UNSIGNED32
Default value	(Device profile dependent)

d. 2nd TPDO mapping (analog inputs)

This TPDO transmits event-driven the 16-bit values of maximum 4 analog inputs. The default transmission type shall be 255; the default values for inhibit and event timer are 0. By default interrupt source (object 6423_{hex}) is disabled. If one analog input changes its value and object 6423_{hex} is enabled, the PDO shall be transmitted immediately. If an analog interrupt condition is enabled, the PDO shall be transmitted only if the interrupt condition is fulfilled. If more than one interrupt condition is enabled, the PDO shall be transmitted if one of these conditions is fulfilled.

Index	Subindex	Comment	Default value
$1A01_{hex}$	0_{hex}	Number of mapped objects	No
$1A01_{hex}$	1_{hex}	1^{st} object to be mapped	6401 01 10_{hex}
$1A01_{hex}$	2_{hex}	2^{nd} object to be mapped	6401 02 10_{hex}
$1A01_{hex}$	3_{hex}	3^{rd} object to be mapped	6401 03 10_{hex}
$1A01_{hex}$	4_{hex}	4^{th} object to be mapped	6401 04 10_{hex}

The number of mapped objects into the PDO depends on the hardware.

SDO_CMDS.rcp

Name	0	5	6	7
bySDO_Send_Code	... .. .	BYTE#16#2F	BYTE#16#23	BYTE#16#2B
wSDO_Send_Index	... .. .	WORD#16#1A04	Word#16#1804	Word#16#2400
bySDO_Send_SubIndex	... .. .	BYTE#16#0	BYTE#16#01	BYTE#16#01
dwSDO_Send_Value	... .. .	DWORD#16#02	DWORD#16#00000182	DWORD#16#800A

Note : Se référer au manuel de votre coupleur CANopen.

5- Validation du mapping du PDO en programmant, au sous-index du nombre d'objets mappés dans le PDO.

Le nombre d'entrées analogiques qui a été programmé (commande SDO 3- et 4-) est 2 (***Send_Value*** *#16#2*). ***Send_Code*****:** *16#2f* (transfert de 1 octet de données (*Send_Value*).

Send_Index du mapping de TPDO5 : 16*#1A04.* ***Send_Subindex*** *#0.*

B 7.26 Object $1A00_{hex}$ – $1A1F_{hex}$: Transmit PDO Mapping Parameter

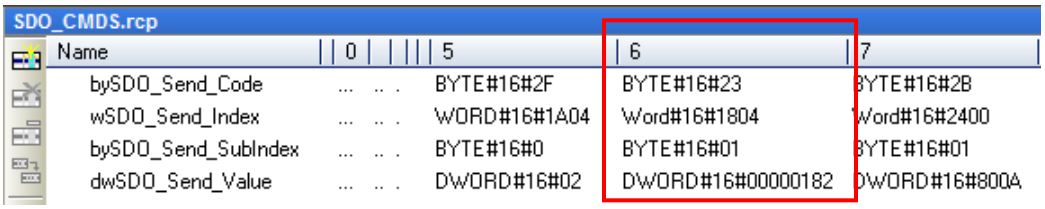
Contains the mapping for the PDOs, which the device can transmit.

a. Object description

Index	$1A00_{hex}$ — $1A1F_{hex}$
Name	Transmit PDO Mapping Parameter
Object	RECORD
Data type	PDO mapping
Category	Conditional; mandatory for each supported PDO

b. Entry description

Subindex	0_{hex}
Description	Number of mapped application objects in PDO
Entry category	Mandatory
Access	RO; RW if dynamic mapping is supported
PDO mapping	No
Value range	0: Deactivated 1 - 64: Activated
Default value	(Device profile dependent)
Subindex	1_{hex} - 40_{hex}
Description	PDO mapping for the n-th application object to be mapped


SDO_CMDS.rcp

Name	0	5	6	7
bySDO_Send_Code	... .. .	BYTE#16#2F	BYTE#16#23	BYTE#16#2B
wSDO_Send_Index	... .. .	WORD#16#1A04	Word#16#1804	Word#16#2400
bySDO_Send_SubIndex	... .. .	BYTE#16#0	BYTE#16#01	BYTE#16#01
dwSDO_Send_Value	... .. .	DWORD#16#02	DWORD#16#00000182	DWORD#16#800A

Note : Se référer au manuel de votre coupleur CANopen.

6- Validation du PDO en programmant son nouveau COB-ID.

Le TPDO5 est mapper au TPDO1 du nœud 2 avec le COB-ID 182 (se référer à la commande SDO 0).

TPDO5 : ***Send_*Index** #16#*1804.* Assigner un identificateur en écrivant 182 (***Send_Value*** *#16#00000182*) au sous-index du COB-ID (ex. : ***Send_Subindex*** *#1*). Dans ***Send_Code***, inscrire *16#23* (transfert de 4 octets de données (*Send_Value*)).

B 7.25 Object 1800_{hex} – $181F_{hex}$: Transmit PDO Communication Parameter

Contains the mapping for the PDOs the device is able to transmit.

a. Object description

Index	1800_{hex} — $181F_{hex}$
Name	Transmit PDO Communication Parameter
Object	RECORD
Data type	PDO CommPar
Category	Conditional; mandatory for each supported PDO

b. Entry description

Subindex	0_{hex}
Description	Largest subindex supported

Subindex	1_{hex}
Description	COB-ID used by PDO
Entry category	Mandatory
Access	RO; RW if COB-ID can be configured
PDO mapping	No
Value range	UNSIGNED32 (Figure 65)
Default value	Index 1800_{hex}: 180_{hex} + Node-ID, Index 1801_{hex}: 280_{hex} + Node-ID, Index 1802_{hex}: 380_{hex} + Node-ID, Index 1803_{hex}: 480_{hex} + Node-ID, Index 1804_{hex} - $18FF_{hex}$: Disabled

SDO_CMDS.rcp

Name	0	6	7	8
bySDO_Send_Code	... .. .	BYTE#16#23	BYTE#16#2B	BYTE#16#2B
wSDO_Send_Index	... .. .	Word#16#1804	Word#16#2400	Word#16#2400
bySDO_Send_SubIndex	... .. .	BYTE#16#01	BYTE#16#01	BYTE#16#02
dwSDO_Send_Value	... .. .	DWORD#16#00000182	DWORD#16#800A	DWORD#16#800A

Note : Se référer au manuel de votre coupleur CANopen.

7- Programmation de l'entrée analogique #1 en entrée de courant 4-20mA (0b1000 0000 0000 1010 → 0x800A, ***Send_Value*** :#16#800A).

Dans ***Send_Code***, inscrire *16#2B* (transfert de 2 octets de données (*Send_Value*)).

Send_Index #16#*2400*. ***Send_Subindex*** *#1*

8- Programmation de l'entrée analogique #2 en entrée de courant 4-20mA (***Send_Value*** #16#800A). Dans ***Send_Code***, inscrire *16#2B* (transfert de 2 octets de données (*Send_Value*)).

Send_Index #16#*2400*. ***Send_Subindex*** #2.

B 16.3 Object 2400_{hex}: AIP Range

Determines the input range for a corresponding analog input, see AIP module-specific data sheet.

a. Object description

Index	2400_{hex}
Name	AIP Range
Object	ARRAY
Data type	UNSIGNED16
Category	Manufacturer

b. Entry description

Subindex	0_{hex}
Description	Number of analog inputs
Access	RO
Entry category	Mandatory
PDO mapping	No
Value range	0-254
Default value	0
Subindex	1_{hex}
Description	Analog Input 1 Range
Access	RW
Entry category	Optional
PDO mapping	No
Value range	UNSIGNED16
Default value	No
...	...
Subindex	**(N) = Number of analog inputs**
Description	N = Number of analog inputs
Access	RW
Entry category	Optional
PDO mapping	No
Value range	UNSIGNED16
Default value	No

AIP module-specific datasheet

One process data output word is available for the configuration of each channel.

Set bit 15 of the corresponding output word to 1 to configure the terminal. If bit 15 = 0 the preset configuration is active.

Bit 15:

Code	Configuration
0	**Default**
1	Configuration data

Bit 9 and bit 8:

Code	Filter
00	**16-sample mean value (default)**
01	No filter
10, 11	Reserved

Bit 5 and bit 4:

Code	Format
00	**IB IL (15 bits) (default)**
01	IB ST (12 bits)
10	IB RT (15 bits)
11	Standardized display

Bit 3 to bit 0:

Code	Measuring Range (Voltage)
0000	**0 V to 10 V (default)**
0001	± 10 V
0010 through 0111	Reserved

Bit 3 to bit 0

Code	Measuring Range (Current)
1000	0 mA to 20 mA
1001	±20 mA
1010	4 mA to 20 mA
1011 to 1111	Reserved

Set all reserved bits to 0.

NOTE : Il est à noter que la recette peut être exportée/importée en format CSV pour en faire l'édition. Les étapes 1 à 6 peuvent être répétées pour la programmation d'autres PDO ou d'autres entrées dans le PDO ou les étapes 7 ou 8 pour configurer d'autres entrées analogiques.

5.4.5. Programme de gestion du CANopen

Le programme CANopen_Init réalise les étapes suivantes (iCAN_Step) :

0- Remise à zéro de tous les nœuds afin de retourner à la configuration par défaut;

1- Requête du statut du nœud avec une requête RTR NMT Node Guard.

2- Vérifier si le nœud est opérationnel et passer à l'étape 4 si c'est le cas.

3- Faire la gestion du time-out basée sur la temporisation T_blink_AI dans le cas où il n'y a pas de réponse : boucler à l'étape 1 si c'est le cas.

```
// Time base  for time-out management
T_blink_AI(True, T#1000ms);  // Period between each Analog input scan
r_trig_AI(T_blink_AI.Q);     // Edge detect
T_blink_DI(True, T#60000ms); // Period between DI integrity Scans
r_trig_DI( T_blink_DI.Q);    // Edge detect
T_HB(True, T#10000ms);       // Period between heart beat
r_trig_HB(T_HB.Q);           // Edge detect

// Init sequence
CASE iCAN_Step OF
    0:
      // Reset all CAN Open nodes
      bRESET_CANOpen_Nodes:=True;
      // Go to next step
      iCAN_Step:= iCAN_Step + 1;

    1:
      // Request the Node 1 status
      bNode_Guarding_Node1:=True;
      byScratchPad:=0;
      iCAN_Step:= iCAN_Step + 1;

    2:
      // Verify if Node 1 is pre-operational
      byScratchPad:= AND_MASK(byNodeGuardAnswer,NodeGuardAnswerMask);
      if (byScratchPad = Preoperational) then
          iCAN_Step:= iCAN_Step + 2;  // Step 4
      else
          iCAN_Step:= iCAN_Step + 1;  // Step 3
      end_if;

    3:
      // Wait until Node 1 is pre-operational (loop to 1)
      if r_trig_AI.Q then iCAN_Step:=iCAN_Step-2;
      end_if;
```

L'étape 4 fait la préparation des commandes SDO à partir de la recette.

```
4:
  // Prepare the SDO commands initialization.
  // SDO commands are stored in SDO_CMDS recipe
  // Send_index in first column (0) contains
  // the quantity of SDO commands to send
  ApplyRecipeColumn ('SDO_CMDS.rcp', 0);
  diSDO_CMDS_QTY:=ANY_TO_DINT(wSDO_Send_Index);        // Number of SDO commands
  diSDO_CMDS_PTR:=1;
  iCAN_Step:= iCAN_Step + 1;                          // SDO commands pointer
```

Les étapes suivantes font l'envoi et la réception des commandes SDO :

5- Envoi de la commande SDO. Il est à noter que la valeur de la commande (double-mot) est assignée à 4 octets. Cette conversion n'est pas nécessaire dans les versions 1.7 et plus de l'atelier RightWON Configuration Suite. À cet effet, la variable dwSDO_Send_Value peut être assignée directement à la requête 601 (offset 4).

6- Vérification de la réponse à la requête SDO. Bouclage à l'étape 5 tant qu'il y a des commandes à envoyer. S'il y a une erreur dans la réponse, on passe à l'étape 100 et on indique l'erreur aux autres applications (iCANStatus)

```
5:
  // Load the SDO command an send it
  ApplyRecipeColumn ('SDO_CMDS.rcp', diSDO_CMDS_PTR); // Load the command
  // Conversion from DWORD to Byte (overcome driver problem in version 1.6)
  bySDO_Send_Value00:=lobyte(loword(dwSDO_Send_Value));
  bySDO_Send_Value01:=hibyte(loword(dwSDO_Send_Value));
  bySDO_Send_Value02:=lobyte(hiword(dwSDO_Send_Value));
  bySDO_Send_Value03:=hibyte(hiword(dwSDO_Send_Value));
  bSDO_Send_CMD:=True;  // Send the command
  iCAN_Step := iCAN_Step + 1; // Next step

6:
  // Wait for the SDO response receive flag
  if bSDO_Rec_Flag = True then
      // Validate the answer
      bSDO_Rec_Flag:=False;
      byScratchPad:= AND_MASK(bySDO_Rec_Code,SDOAnswerMask);
      if byScratchPad = SDODownloadAnswer then
          // Valid answer : Execute next command if any
          diSDO_CMDS_PTR:= diSDO_CMDS_PTR + 1; // Next command
          if diSDO_CMDS_PTR > diSDO_CMDS_QTY then
              iCAN_Step := iCAN_Step + 1;    // Valid response
          else
              iCAN_Step := iCAN_Step - 1 ;   // Loop for next
          end_if;
      else
          // Invalid answer.  Indicate the error
          iCANStatus:= CANInitError;  // Initialization error
          iCAN_Step:=100;
      end_if;
  end_if;
```

Les étapes 7 à 11 réalisent les actions suivantes :

7- Démarrage des nœuds

8- Requête RTR de lecture du statut du nœud 1

9- Vérifier que le nœud est opérationnel et passer à l'étape 11 dans ce cas.

10-Gestion du time-out de la réponse du nœud. Passe à l'étape 8 dans le cas où le nœud n'est pas opérationnel.

11-Requête de lecture des entrées numériques et préparation à l'envoi périodique des TPDO.

```
7:
// Start CANbus nodes
   bSTART_CANOpen_Nodes := True; // Go in operational mode
   iCAN_Step := iCAN_Step + 1;   // Next step

8:
// Request the Node 1 status
  bNode_Guarding_Node1:=True;
  iCAN_Step:= iCAN_Step + 1;

9:
// Verify if Node 1 is Operational
  byScratchPad:= AND_MASK(byNodeGuardAnswer,NodeGuardAnswerMask);
  if (byScratchPad = Operational) then
      iCAN_Step:= iCAN_Step + 2;  // Step 11
  else
      iCAN_Step:= iCAN_Step + 1;  // Step 10
  end_if;

10:
// Wait until Node 1 is operational (loop to 8)
  if r_trig_AI.Q then iCAN_Step:=iCAN_Step-2;
  end_if;

11:
// Force a digital input scanning
   bPoll_TPDO1 := True;
   byNodeGuardAnswer:=0;
   bHeartBeatRcv:=False;
   iCANStatus := CANRunning;   // Initialization done!
   iCAN_Step := iCAN_Step + 1; // Next step
```

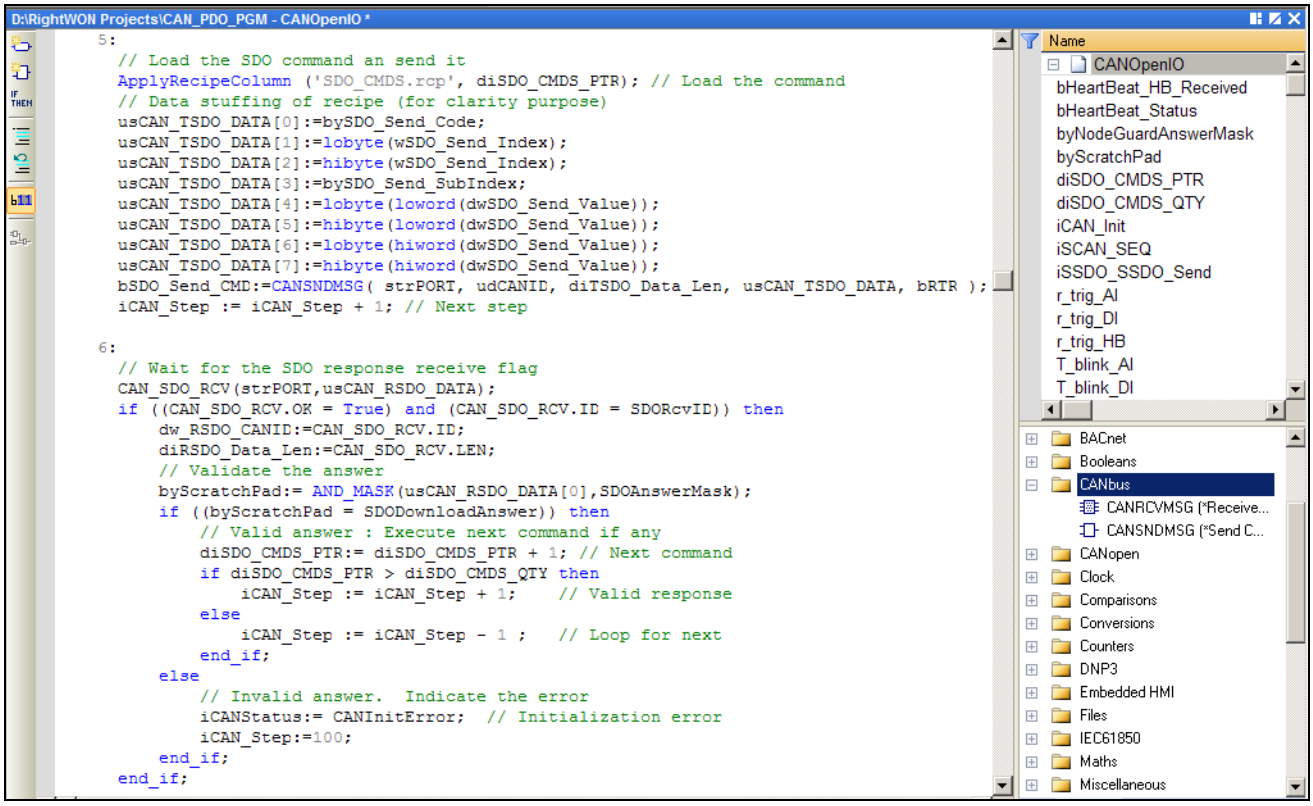
À l'étape 12, les requêtes de lecture sur demande (RTR) des entrées numériques (TPDO1) et des entrées analogiques (TPDO2 et TPDO5) sont effectuées périodiquement. En plus, une commande de lecture périodique du statut du nœud est effectuée afin de détecter les erreurs de communication entre le RightWON et le coupleur CANopen.

```
    12:
    // Periodic scanning of inputs
       bPoll_TPDO1 := r_trig_DI.Q;
       bPoll_TPDO2:= r_trig_AI.Q;
       bPoll_TPDO5:= r_trig_AI.Q;

    // Send the heart beat
       bNode_Guarding_Node1:=r_trig_HB.Q;

    // Node Heartbeat
       // Send the heartbeat at rising edge of blink timer
       if bNode_Guarding_Node1 then
          byNodeGuardAnswer:=1;  // Indicate that the HB is sent
       end_if;
       // Verify the validity of node status if a message is received
       if bHeartBeatRcv then
          bHeartBeatRcv:=False;
          byScratchPad:= AND_MASK(byNodeGuardAnswer,NodeGuardAnswerMask);
          if (byScratchPad <> Operational) then
             // Invalid answer.  Indicate the error
             iCANStatus:=CAN_BadNode;  // Initialization error
             iCAN_Step:=CAN_BadNode;
          end_if;
          // Otherwise check the time-out
          else
             // No answer time-out management
             if ((not(T_HB.Q))and byNodeGuardAnswer = 1) then
                 iCANStatus:= CAN_NoHBAnswer;
                 iCAN_Step:=CAN_NoHBAnswer;
             end_if;
          end_if;

     ELSE
       iCAN_Step := iCAN_Step;
END_CASE;
```

5.5. Utilisation des fonctions CANSNDMSG et CANRCVMSG

Les fonctions **CANSNDMSG** et **CANRCVMSG** peuvent être utilisées pour faire l'écriture et la lecture de données sur le CANbus à partir d'un programme d'application plutôt qu'à partir du configurateur de bus de terrain. Le projet CAN_PDO_PGM contient le même exemple que le projet CAN_PDO_Example. Par contre, les étapes 5 et 6 font l'envoi des commandes SDO avec la fonction CANSNDMSG et la réception des réponses avec la fonction CAMRCVMSG plutôt que de faire les échanges avec le configurateur de bus de terrain. Les commandes sont envoyées par un tableau (*usCAN_TSDO_DATA[]*) assigné à partir de la recette modifiée SDO_CMDS qui inclut le nombre d'octets à envoyer (*diTSDO_Data_Len*).

Il est à noter que les réponses paramétrées par la configuration de bus de terrain sont interceptées et ne sont pas acheminées à la fonction CANRCVMSG. De plus, il faut configurer le FIFO associé au pilote dès que les fonctions **CANSNDMSG** et **CANRCVMSG** sont utilisées.



Vizimax Inc.
2284 de la Province
Longueuil, Québec
Canada J4G 1G1

Tel. (450) 679-0003
Fax : (450) 679-9051
Sales@vizimax.com
www.vizimax.com