Guide de démarrage Modbus

Sommaire

Historique du projet

Date	Auteur	Modifications
30.04.2009	TCS / sr	Rédaction de la documentation (version 1) et du projet pour PG5 1.4.300
3.07.2009	TCS / sr	Projet et documentation modifiés et adaptés (version 2)
03.12.2009	TCS / sdu	Meilleure explication du Media Mapping et UID (version 3)

1 Introduction

Ce document est destiné à faciliter les premières utilisations du Saia Modbus. Allié au projet PG5 correspondant, il peut servir de fil conducteur pour la réalisation d'une communication Modbus.
Les informations contenues dans cette documentation sont extraites des manuels et de l'aide en ligne correspondants et sont destinées à vous faciliter l'entrée en matière. Pour de plus amples informations, veuillez consulter les documents correspondants (voir chapitre « Références »).

2 Matériels et logiciels nécessaires

Matériels
Ce projet est configuré pour la combinaison matérielle suivante :

- PCD3.M5540 en tant que Client IP et/ou RS
  Firmware 1.10.16 ou supérieur
- PCD3.M3330 en tant que Serveur IP
  Firmware 1.10.16 ou supérieur
- PCD2.M5540 en tant que Serveur RS
  Firmware 1.10.16 ou supérieur
- Câble Ethernet (CAT5) pour la connexion entre le PCD3 Client (IP) et le PCD3 Serveur (IP)
- Câbles pour la connexion des interfaces RS 485
- Un câble USB (longueur maximale 1,8 m) pour la programmation des PCD

Logiciels
Les logiciels suivants avec leurs licences valides sont nécessaires pour la programmation des PCD :

- PG5 1.4.300
  Pour la programmation des PCD. (La bibliothèque HLK a été utilisée pour la fonction d'horloge, mais ne fait pas partie des conditions pour le fonctionnement de la communication Modbus.)
- Bibliothèque Modbus Saia (min. $1.0.0.14)

Naturellement, il est aussi possible d'exploiter ce projet avec d'autres matériels. A cet effet, des adaptations spécifiques de la configuration devront être réalisées en fonction des matériels (configuration matérielle dans PG5, paramétrage des logiciels dans PG5, mémoire disponible dans Fupla et réglages correspondants pour la communication entre les PCD).
La bibliothèque Modbus Saia n'est compatible qu'avec les nouveaux systèmes tels que PCD3 et PCD2.M5xxx.

3 Bases Modbus

Ce chapitre est destiné à fournir un bref aperçu de Modbus. Il décrit les applications Modbus, son historique, ses particularités et les différences par rapport au S-bus. Le mappage des ressources PCD est également expliqué brièvement. De plus, le chapitre contient la description des différences entre la bibliothèque Modbus Saia et la bibliothèque d'Engiby existante.

3.1 Applications typiques

3.1.1 Sériel

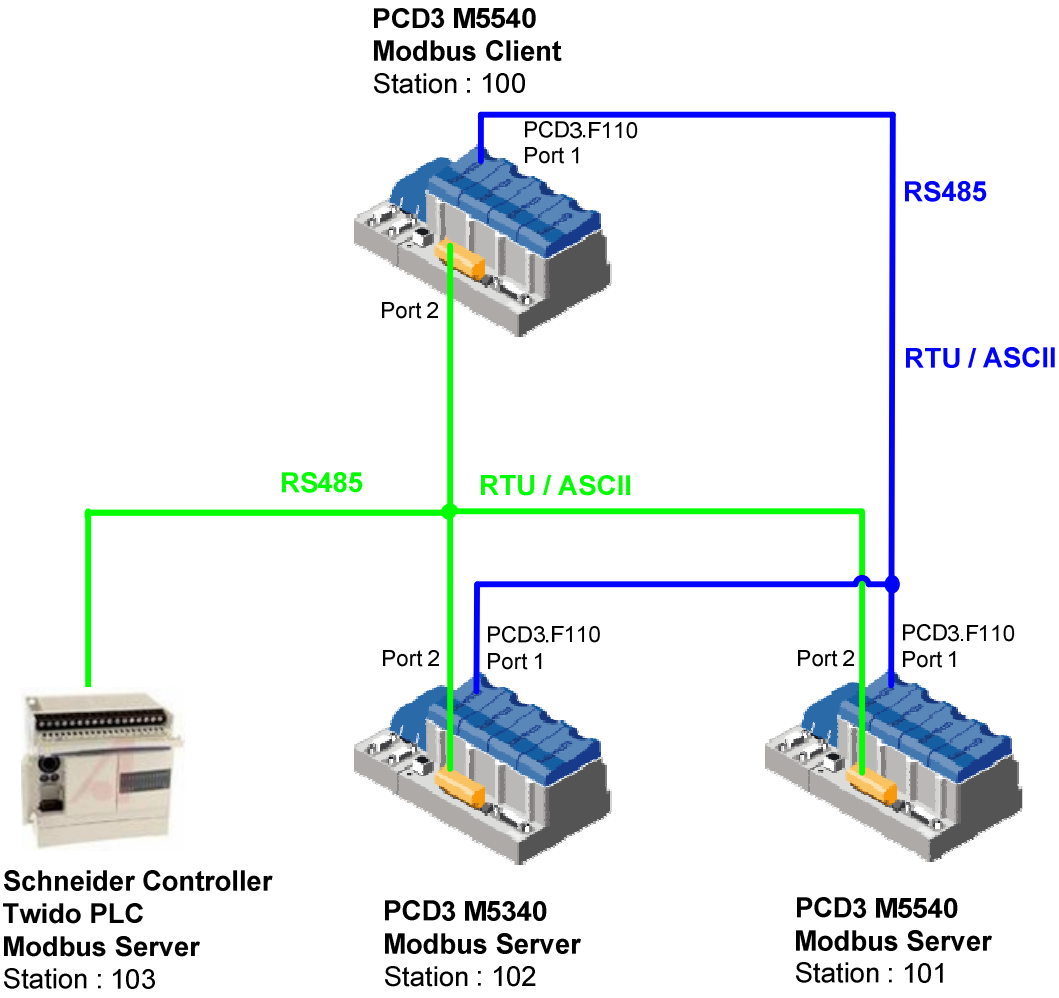


Figure 3.1.1.1 Structure sérielle

3.1.2 Ethernet

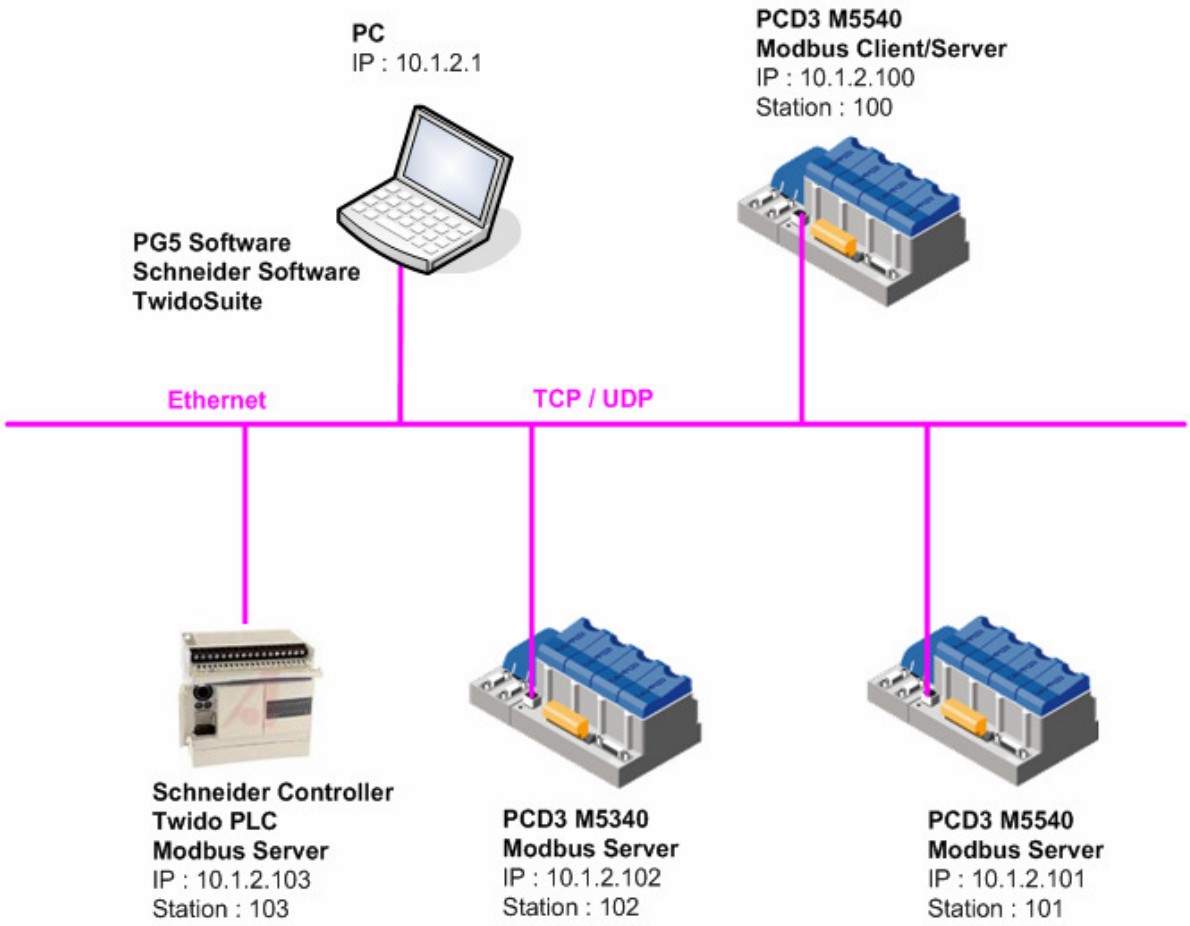


Figure 3.1.2.1 Structure Ethernet

3.2 Historique

Le Modbus a été introduit en 1979 par la société Modicon, l'actuelle société Schneider. Il s'agit d'un bus à maître unique. Le protocole Modbus est devenu au fil du temps un standard pratique qui est utilisé par de nombreux fabricants.
Protocoles de communication Modbus :

- Modbus RTU : à codage binaire avec des interruptions entre les télégrammes. Ce type de communication est efficace, mais sensibles aux retards lors du transfert de caractères.
- Modbus ASCII : permet une interprétation / lecture facile par les êtres humains. Il utilise des caractères d'amorçage et d'arrêt. La longueur des télégrammes est env. 2 fois supérieure à celle du Modbus RTU.
- Modbus TCP/UDP : est utilisé en tant que réseau client/serveur. Implémentation de Modbus sur Ethernet (TCP/IP et UDP/IP). Ceci permet la réalisation d'une topologie de réseau à maîtres multiples.

3.3 Comparaison S-bus ⇔ Modbus

Bien que le S-Bus sériel et le Modbus RTU / ASCI soient disponibles sur les mêmes couches physiques, ils présentent certaines différences. Les protocoles sont comparés dans les tableaux ci-dessous.

- Un point important est le fait qu'un serveur S-bus (esclave) dispose toujours d'une adresse, alors qu'un serveur Modbus (esclave) peut avoir plusieurs UID.
- Dans le S-bus, tous les médias (registres, entrées, indicateurs, etc.) disposent d'une adresse fixe. Sur le Modbus cependant, un mappage spécifique peut être défini pour chaque UID (Unique Identifier).
- La longueur des registres Modbus est de 16 bits. La longueur des registres PCD est de 32 bits.

3.3.1 Comparaison S-bus sériel ⇔ Modbus RTU/ASCII

Critère	S-bus sériel	Modbus RTU/ASCII
Interface physique	RS 485, RS 232, RS 422, boucle de courant…	Idem, le modem sériel n'est pas compatible avec notre implémentation
Nombre maximal de maîtres	1	Idem
Nombre maximal d'esclaves	252	247
Adresses par esclave	1 adresse S-bus	Jusqu'à 9 identifiants uniques (UID)*
Adresse de Broadcast	255	0
Checksum	CRC 16	Idem
1 bit de données	Entrée / Sortie, Flag	Discret Input (DI, lecture uniquement) Coil (lecture et écriture)
16/32 bits de données	Registre (32 bits) DB (32 bits)	Input Register (IR, 16 bits, lecture uniquement) Holding Register (HR, 16 bits, lecture et écriture)
Première adresse de données	0	1
Vitesses de transfert compatibles	300-115000	1200-115000**
Nombre de ports exploitables avec le protocole	Identique au nombre d'interfaces série.	Maximum 10
Mélange de protocoles sur un PCD	Oui	Idem
Longueur maximale des télégrammes (données)	128 octets	253 octets
Compatible avec les modules PCD3.F2xx	Oui	Pas encore compatible
Influence sur les performances de la commande.	Les performances diminuent avec chaque port configuré, en fonction de la vitesse de transfert.	Idem

* pour notre implémentation, plus UID 0 par défaut

** en fonction des commandes compatibles

3.3.2 Comparaison S-bus Ethernet⇔ Modbus TCP/UDP

Critère	S-bus Ethernet	Modbus TCP/UDP
Interface physique	Ethernet	Idem
Connexion	Uniquement UDP	TCP ou UDP
Port standard	5050	502
Nombre maximal de serveurs par réseau	Plusieurs	Idem
Nombre maximal de clients par réseau	Plusieurs	Idem
Nombre maximal de connexions de clients par réseaux	Aucune restriction, puisqu'elles sont traitées de manière séquentielle	10
Nombre maximal de connexions de serveurs par réseaux	Aucune restriction, puisqu'elles sont traitées de manière séquentielle	26 moins le nombre des connexions clients
Adresses par esclave / serveur	1 adresse S-bus	Jusqu'à 9 identifiants uniques (UID)*

* pour notre implémentation, plus UID 0 par défaut

Différence importante : Sur le S-bus, un canal de communication est toujours affecté à un port physique. Un canal Modbus se compose d'un port et d'un protocole. Ce qui signifie qu'il est possible de définir plusieurs canaux avec différents protocoles sur un port Ethernet.

Par ex. TCP sur 502, UDP sur 502, TCP sur 503 et ASCII sur le port sériel 2.

3.4 Particularités Modbus et Mapping

3.4.1 Maître/Esclave ⇔ Client/Serveur

Le Modbus sériel est orienté Maître/Esclave, tandis que le Modbus TCP/UDP est orienté Client/Serveur. Pour plus de simplicité, la désignation Client/Serveur est utilisée dans toutes les Fbox. Dans ce cas, Client = Maître et Serveur = Esclave.

3.4.2 16 bits ⇔ 32 bits

Modbus est orienté 16 bits (Word). Le transfert de valeurs de 32 bits est courant, bien que non standardisé, ni pour les valeurs entières, ni pour les valeurs en virgule flottante. De ce fait, on trouve aussi plusieurs voies de mappage de valeurs à 32 bits sur un Holding Register et vice-versa.

Afin de couvrir tous ces cas, nous assurons la compatibilité avec l'ensemble des méthodes de mappage et des processus nécessaires, tels que la permutation de mots et la conversion. Lorsque des données 32 bits sont échangées par un produit tiers, les paramètres de mappage suivant doivent être définis :

- Signé / non signé ?
- Mot inférieur en premier ou en dernier ?
- En cas de virgule flottante : IEEE ou FFP

3.4.3 Offset

Les ressources Modbus sont incrémentées à compter de 1, tandis que les registres PCD débutent à 0. Ce décalage peut provoquer des problèmes, puisqu'il peut être à l'origine de la lecture ou de l'écrite d'un mot erroné. Les paramètres de décalage devront donc être contrôlés des deux côtés. Les Fbox permettent de configurer un offset.

3.4.4 Qu'est-ce qu'un UID

Un UID (Unique Identifier) est comme une adresse S-Bus. Par ailleurs chaque station peut avoir jusqu'à 9 UIDs. Dans un réseau sériel, chaque UID existe qu'une fois. Chaque UID a son propre Mapping. Les UID peuvent être accédés à partir de tous les ports de communication Modbus (TCP, UDP, sériel).

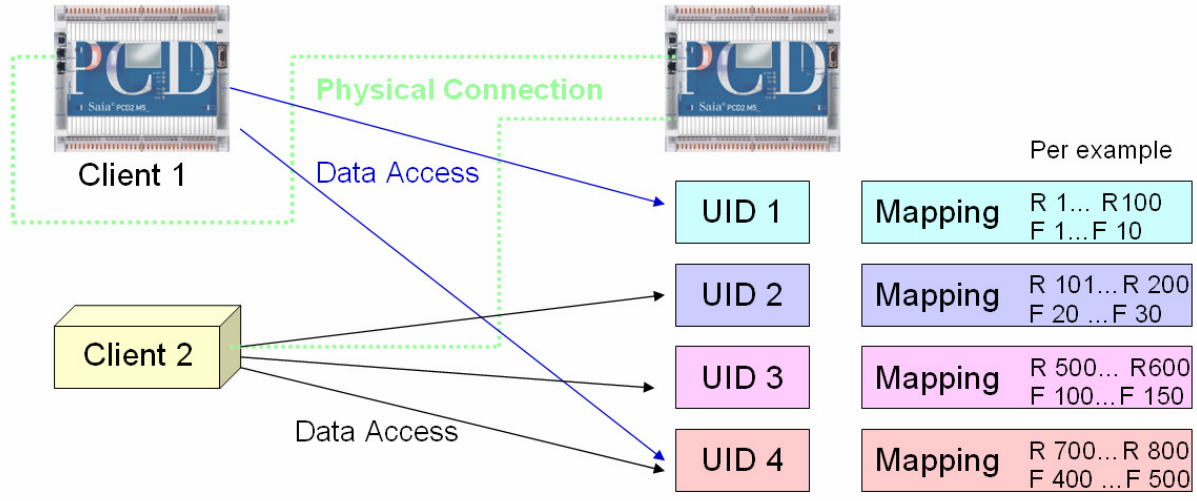


Figure 3.4.4.1 UID

3.4.5 A quoi sert le Mapping?

L'UID permet l'accès aux ressources définies par le Mapping. Pour le Server, les Fbox suivantes sont disponibles.

Figure 3.4.5.1 Fbox Mapping

Le Client peut accéder aux ressources avec les Fbox suivantes.

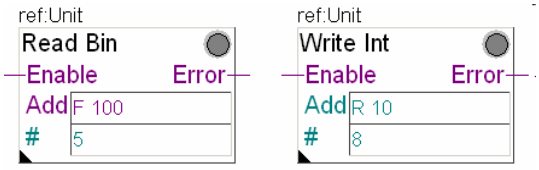


Figure 3.4.5.2 Fbox Read/Write

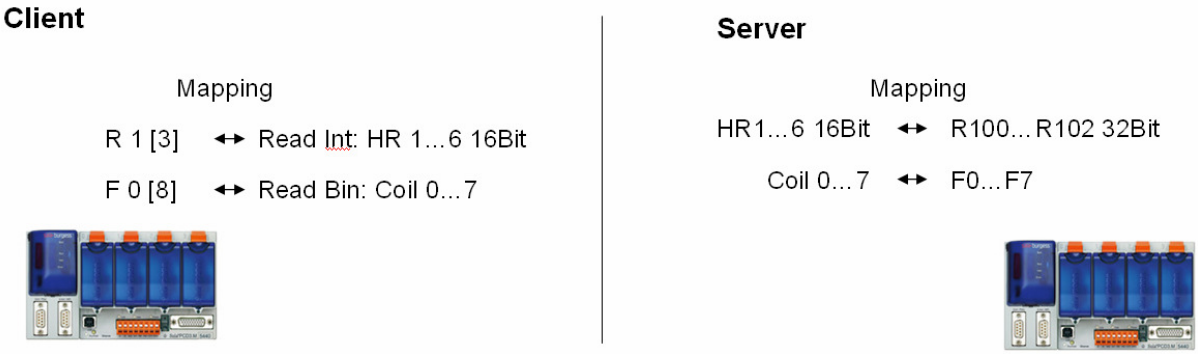


Figure 3.4.5.3 Fbox Read/Write

3.4.6 Mappage et UID

Chaque serveur PCD peut disposer jusqu'à 9 UID Modbus. Celles-ci sont toutes configurables par l'utilisateur. Un mappage séparé peut être défini pour chaque UID. Chaque UID permet l'affectation de jusqu'à 10 zones de mappage. Le mappage par défaut peut toujours être utilisé à la place du mappage configurable par l'utilisateur. Dans ce cas, toutes les ressources sont disponibles. S'il existe d'autres mappages, il est impératif de veiller à ne pas provoquer de collisions.

Les UID s'appliquent toujours à tous les ports Modbus (sériels et TCP/UDP). Dans un réseau sériel, chaque UID doit être univoque.

Les UID peuvent être reconfigurées en Runtime.

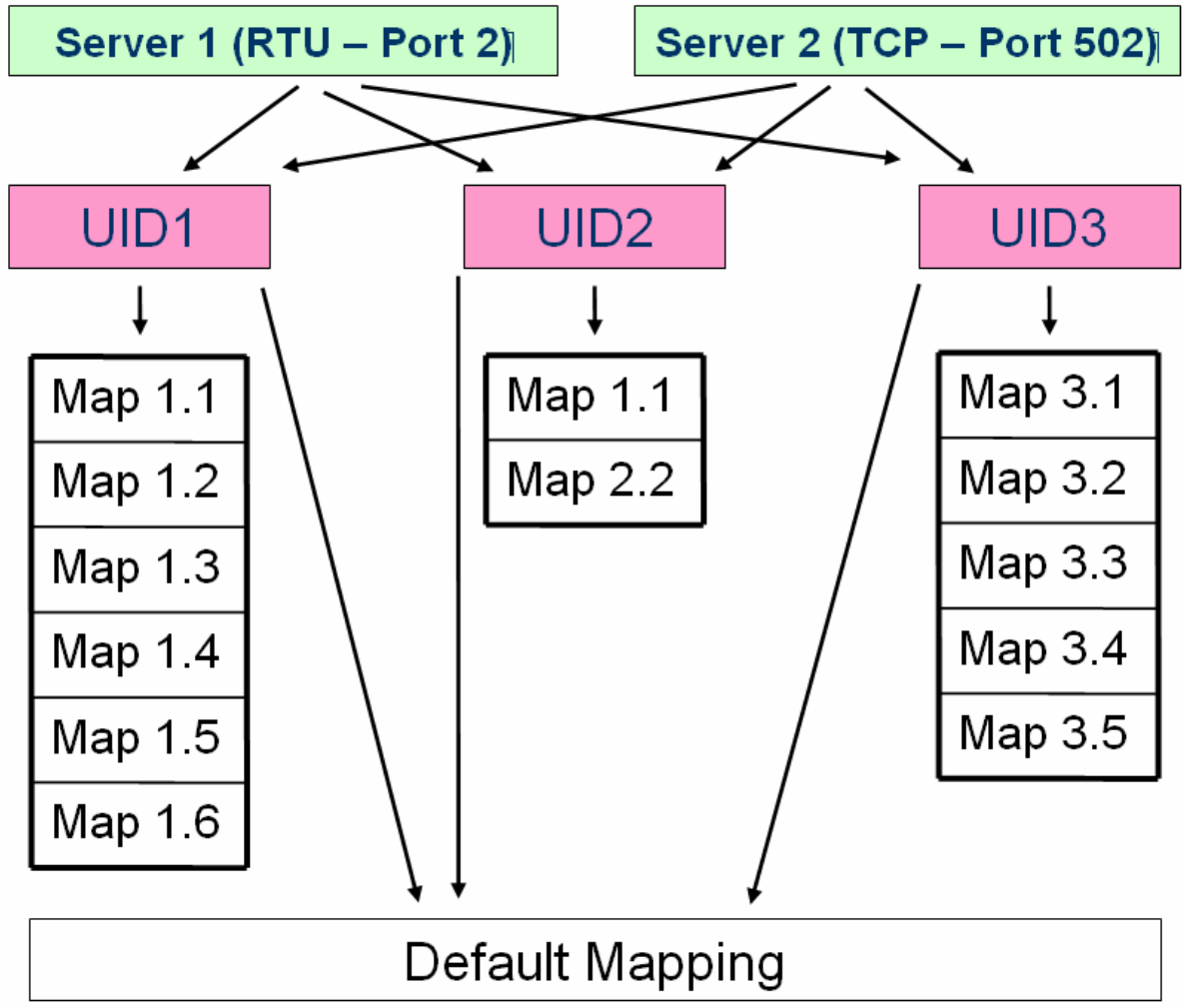


Figure 3.4.6.1 Mappage

Si aucun mappage n'est défini, le mappage par défaut reste actif. Toutefois, dès qu'un mappage est établi pour une UID, celui-ci est prioritaire.

L'UID 0 est l'UID par défaut pour le Modbus sériel. Cette UID est toujours disponible et peut servir à transmettre des télégrammes de Broadcast.

Lors de la communication TCP/UDP, l'UID par défaut est également toujours active, mais aucun mappage par défaut n'a été défini. Les interrogations transmises à une UID inexistante sont toujours traitées par l'UID par défaut. Dès qu'un mappage est

défini pour l'UID par défaut, celui-ci est utilisé par l'UID par défaut ainsi que pour toutes les UID non configurées.

3.4.7 Définition du canal de communication

Un canal Modbus est toujours défini par un port et un protocole. Par exemple: TCP sur le Port 502 ou ASCII sur le port sériel 2.

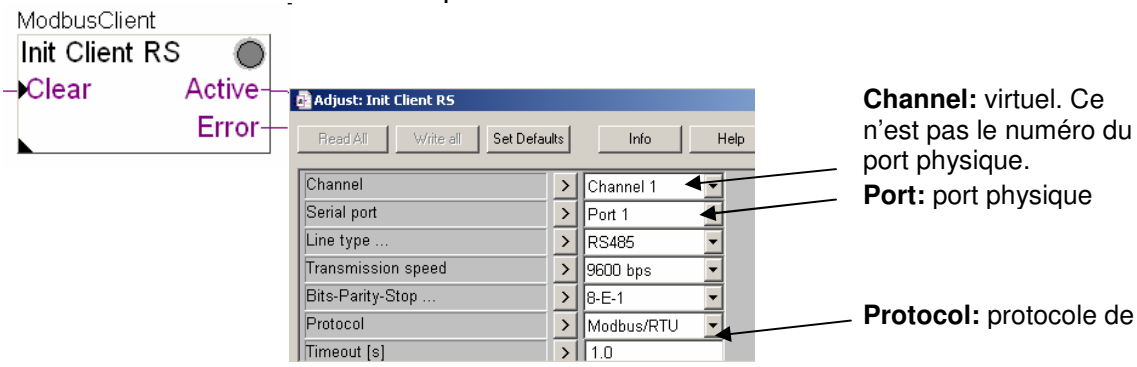


Figure 3.4.7.1 Définition du canal

Chaque canal ne peut être défini qu'une fois. C'est possible de définir jusqu'à 4 canaux sur un PCD Server.

3.4.8 Connexions

Lorsqu'un client requiert des données auprès d'un serveur, une connexion est établie automatiquement. L'utilisateur n'a pas besoin de s'occuper de la connexion et de la déconnexion.
Néanmoins, il peut être important de savoir à quel moment une connexion est établie ou coupée, puisque le nombre de connexions est limité. Les détails figurent dans le manuel Modbus 26/866.

3.4.9 Holes

Lorsque des valeurs à 32 bits sont mappées sur des registres, deux variantes sont disponibles au choix. Il est possible de travailler avec des Holes (trous). Les données sont alors plus faciles à interpréter, parce que les adresses de registres sur le PCD correspondent aux adresses des Holding Registers sur le Modbus. Dans ce cas, seuls les registres pairs sont utilisés, les registres impairs restant vierges. Sans Holes (trous) le tout est plus compact et économique en ressources, mais l'affectation est un peu plus complexe.

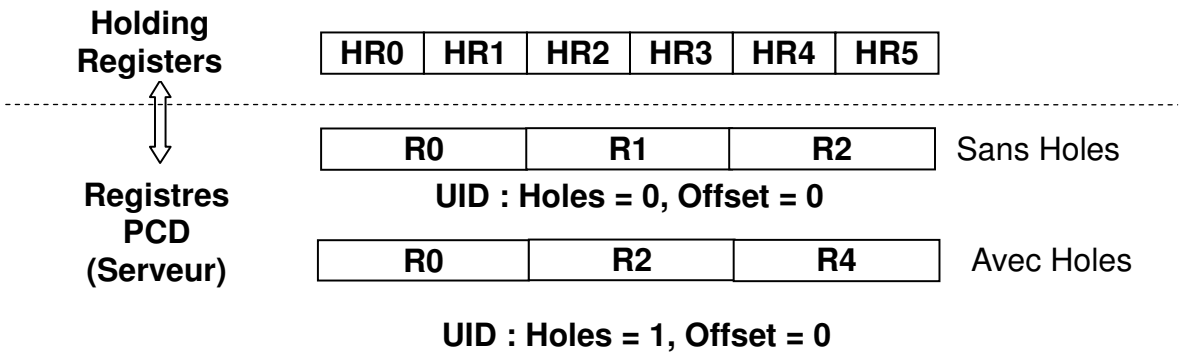


Figure 3.4.9.1 Holes

3.4.10 Programmation du Client

En tant que Client nous devons savoir à quel UID accéder et où les ressources sont mappées.
De plus il est nécessaire de vérifier si ce sont des valeurs binaires 16bit ou 32bit et si il y a un offset et des holes.

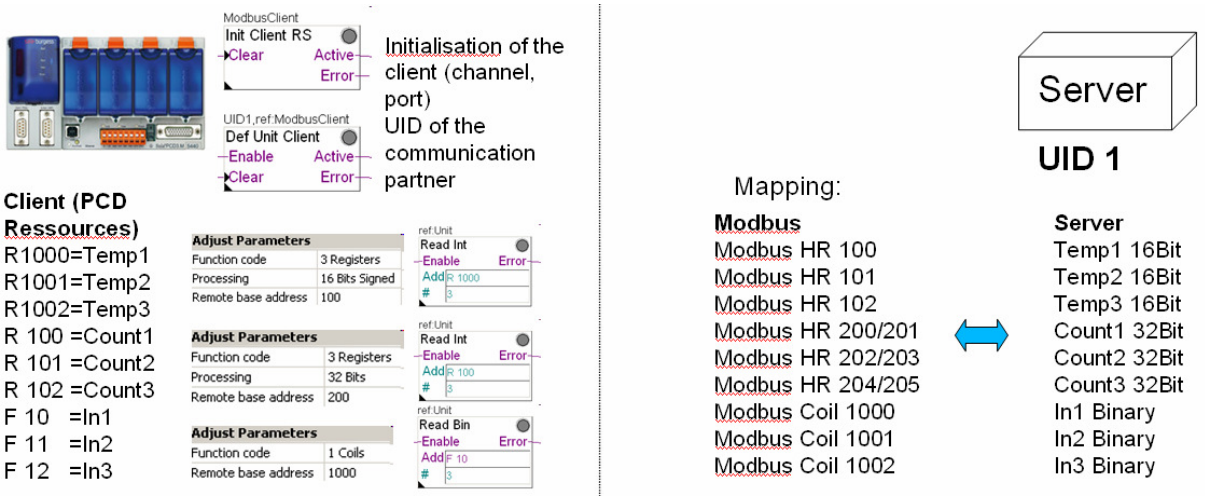


Figure 3.4.10.1 Client

Il est important de réserver la bonne plage mémoire pour les ressources reçues. C'est-à-dire que si l'adresse de base est le registre 1000 et que 3 éléments sont lus, nous devons réserver les registres 1000, 1001 et 1002. PG5 ne reconnaît pas durant la compilation que plus d'un même registre est utilisé. Pour cela il est recommandé d'utiliser un tableau de registres, par exemple R 1000 [3] ou de s'assurer que ces registres ne sont pas utilisés ailleurs.

3.4.11 Programmation du Server

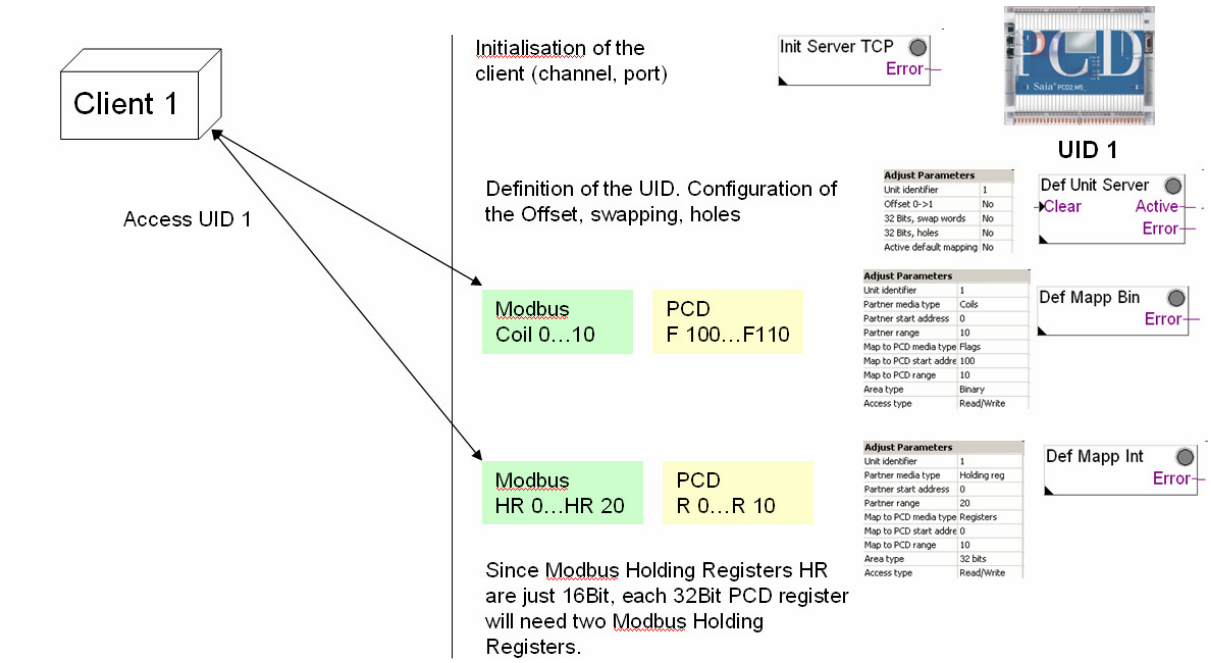


Figure 3.4.11.1 Server

En tant que Server nous fournissons les ressources pour chaque UID avec le mapping.

3.5 Comparaison entre les solutions Saia ⇔ Engiby

Si la bibliothèque Engiby a été utilisée auparavant, il est important que le mappage soit transféré correctement. Certaines désignations sont différentes, ce qui peut prêter à confusion. Vous trouverez ci-dessous une vue d'ensemble des différentes désignations dans les Fbox Engiby et Saia :

FBox Saia	**FBox Engiby**
32 bit swap words NO	LITTLE Endian
32 bit swap words YES	BIG Endian
Holding Register	R
Input Register	InR

Limites à respecter

Résumé des limites à respecter pour le réseau Modbus :

- Au maximum 247 Serveurs (esclaves) par bus pour le Modbus sériel
- Au maximum 10 canaux pour l'ensemble, dont tout au plus 4 canaux Serveurs (port + protocole)
- Au maximum 9 UID personnalisés (+ UID par défaut)
- Au maximum 10 zones de mappage par UID
- La connexion modem sérielle n'est pas compatible
- Fonctions Modbus compatibles :
    - READ_COILS
    - READ-DISC_INPUT
    - READ_HOLD_REG
    - READ_INPUT_REG
    - WRITE_SINGLE_COIL
    - WRITE_SINGLE_REG (16 bits uniquement)
    - WRITE_MULTIPLE_COILS
    - WRITE_MULTIPLE_REGS

4 Description du projet d'exemple

Il y a 3 PCDs dans ce projet. Une communication IP (Modbus TCP) est établie entre le "Client" et le "Modbus_Server" et une communication sérielle RS485 (Modbus ASCII) est établie entre le "Client" et le "Server_RS".

4.1 Echange de données à travers Modbus

Les ressources PCD (R, F, T, C, I ,O…) ne peuvent pas directement être transférées par Modbus. Dans le Modbus, on parle de Holding Registers et de Coils (read/write) ou d'Input Registers et de Discrets Inputs (read only). C'est pour cela qu'il est nécessaire de faire un mapping pour chaque UID. On peut utiliser le mapping par défaut. Ce mapping est le lien entre les ressources PCD et les ressources Modbus. Les détails concernant ce mapping sont dans le manuel Modbus 26/866 et au chapitre 3.4.4 ci-dessus.

4.1.1 Communication IP

Les données suivantes sont échangées entre le PCD “Client“ et le PCD “Modbus_Server“ :
Sur le “Modbus_Server“ l’horloge est transférée sur les registres 100, 101 et 102 (32 bits). Ces registres seront transmis au “Client“ à travers Modbus et stockés dans les registres 100, 101, 102. L’horloge du “Client“ peut maintenant être mise à jour avec l’horloge du “Modbus_Server“.
Les Flags 1000 à 1007 du “Client“ sont transmis à travers Modbus au “Modbus_Server“ sur les Flags 1000 à 1007. Ces Flags sont incrémentés sur le “Client“ et, si la communication fonctionne correctement, il est possible d’observer le changement d’état de ces Flags sur le “Modbus_Server“.
Le mapping suivant est configuré :

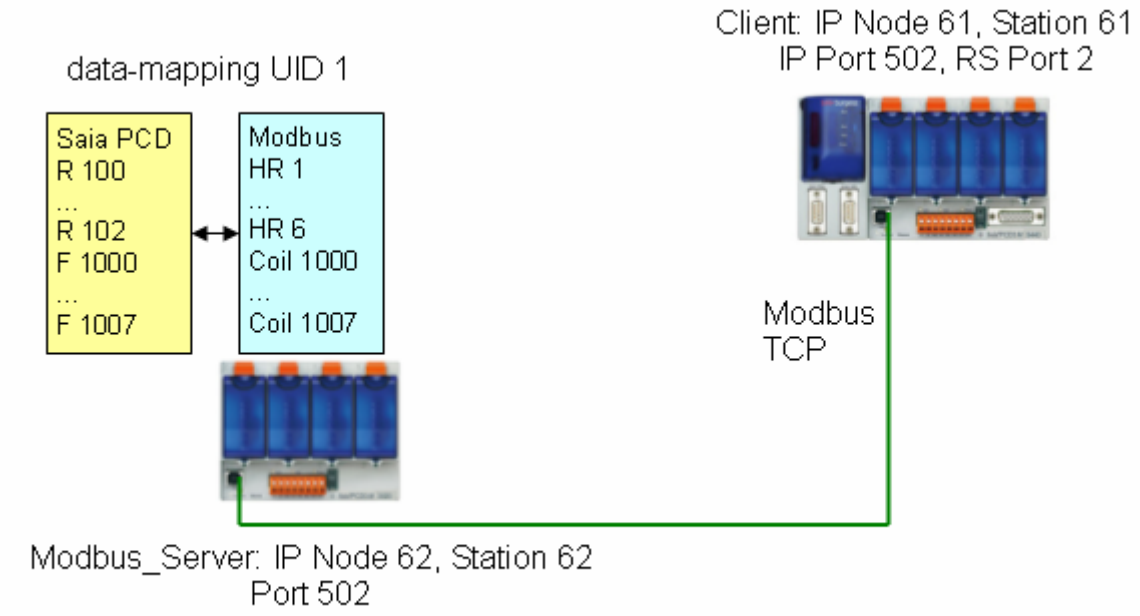


Figure 4.1.1.1 Modbus network-topology

Client IP Saia PCD Medias (R u. F)	**Modbus Holding Register / Coils**	**Server IP Saia PCD Medias (R u. F)**
R 100…R 102 (32 Bit)	<= HR 1…6 (16 Bit) <=	R 100…R 102 (32 Bit)
F 1000…F 1007	=> Coil 1000…1007=>	F 1000…F 1007

4.1.2 Communication RS 485

Les données suivantes sont échangées entre le PCD “Client“ et le PCD “Server_RS“ :
Le “Client“ lit 8 registres et 8 Flags du “Server_RS“. Les registres 0 à 7 et les Flags 0 à 7 sont lus. Les registres sont définis comme 16 bits signed. Il n’y a pas de mapping spécific utilisé. Cela signifie que le mapping par défaut est utilisé. Si on utilise des registres 16 bits signed, cela signifie que tous les registres PCD sont disponibles dans les Holding Registers 1…10000. Les Flags sont disponibles dans les Coils 1…10000.
Le mapping suivant est configuré :

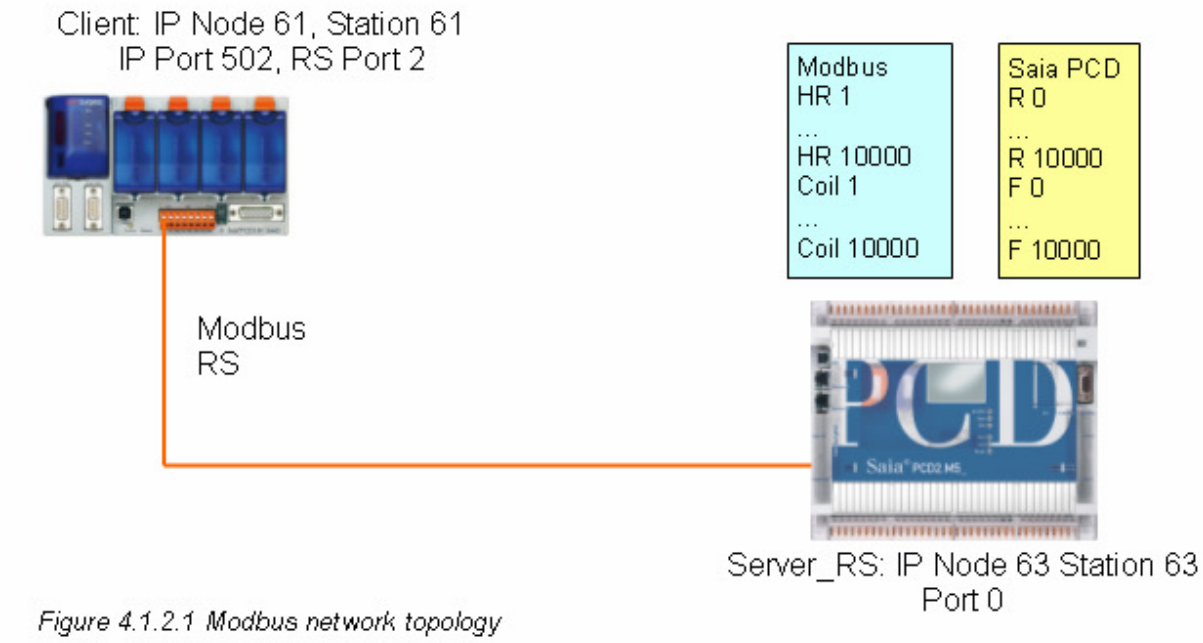


Figure 4.1.2.1 Modbus network topology

Client RS Saia PCD Medias (R u. F)	Modbus Holding Register / Coils	Server RS Saia PCD Medias (R u. F)
R 0…R 7 (16 Bit signed)	<= HR 1…8 (16 Bit) <=	R 0…R 7 (16 Bit signed)
F 100…F 107	=> Coil 1…8=>	F 0…F 7

5 Préparation du projet d'exemple

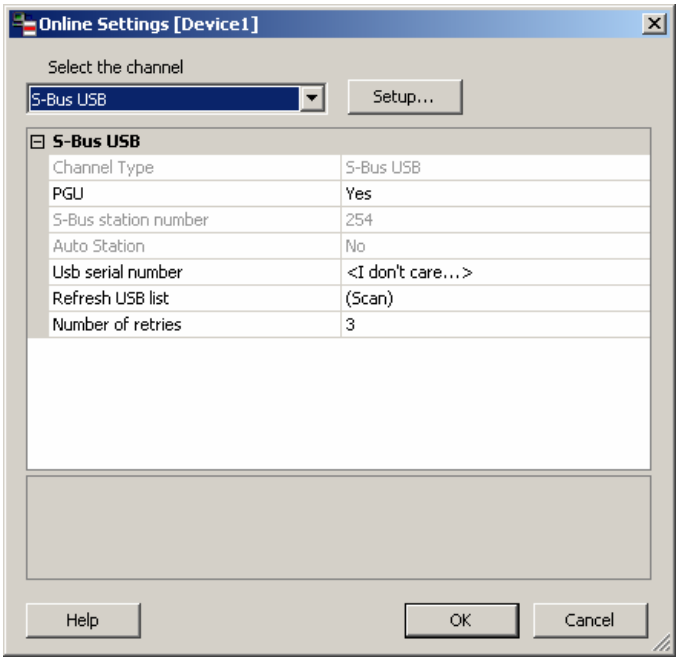
La fonction « Restaurer » du menu « Projet » du gestionnaire de projets PG5 peut être utilisée pour importer le projet dans PG5.

Afin de rendre possible une communication Modbus, au moins deux stations doivent être configurées et programmées.

5.1 Configurer un PCD

Pour préparer le PCD, 3 étapes sont nécessaires :

Etablir une connexion en ligne vers le PCD
Avant de pouvoir établir une connexion, PG5 doit « savoir » par quel moyen/câble il doit accéder au PCD. Ces données sont définies dans l'option « Online Settings » de l'arborescence du projet PG5 :



Sélectionnez comme « Channel » l'option « S-Bus USB », puisque la configuration IP n'est pas encore chargée. L'option PGU doit être activée.

Figure 5.1.1 Online Settings

Après ces paramétrages, le « Online Configurator » permet de vérifier si la communication fonctionne.

Configurer le hardware
La configuration du hardware permet de configurer des paramètres tels que l'adresse IP, l'utilisation de la mémoire et l'activation de l'interrupteur « Run/Stop » du PCD. Les « Hardware Settings » du PCD figurent également dans l'arborescence du projet PG5, directement sous les « Online Settings ».

Avant de poursuivre, veuillez indiquer dans l'onglet TCP/IP une adresse IP et un masque de sous-réseau adaptés à votre réseau et non encore affectés.

Pour charger la configuration dans la commande, il suffit de cliquer sur le bouton « Download» de la fenêtre « Hardware Settings ».
Lors de l'interrogation au sujet de ce qui doit être chargé dans la commande, vous devez également sélectionner l'option « Memory Allocation » pour configurer correctement la mémoire.

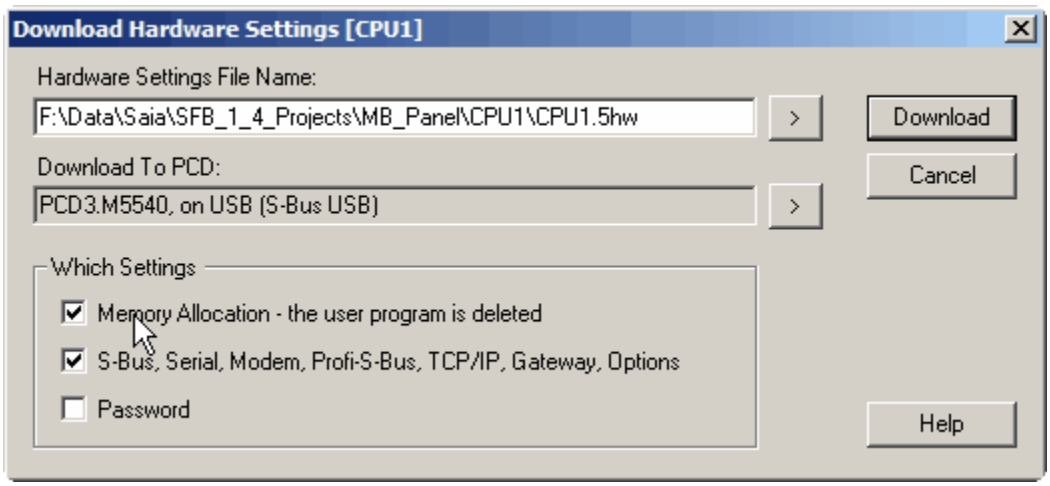


Figure 5.1.2 Download

Si le type exact du PCD n'est pas connu, ou si la configuration existante du hardware ne doit pas être modifiée, il est également possible de sélectionner le bouton « Upload » de la fenêtre « Hardware Settings ». Ainsi la configuration actuelle du PCD est reprise dans le projet PG5.

Les Hardware Settings doivent être adaptées de manière correspondante sur tous les PCD que l'on souhaite utiliser :
Client, adresse S-bus 61, IP Node 61. Ceci est le Client (maître) IP et RS.
Serveur Modbus, Adresse S-bus 62, IP Node 62. Ceci est le Serveur (esclave) IP.
Serveur_RS, Adresse S-bus 63, IP Node 63. Ceci est le Serveur (esclave) RS.

5.2 Configuration étendue

L'exemple est conçu de sorte à disposer d'un client (maître) sur lequel se trouve respectivement un fichier Fupla pour les possibilités de communication IP et RS.

Si la communication Modbus doit être réalisée uniquement sur IP, l'autre fichier doit être désactivé. Si seule la communication RS est souhaitée, le fichier IP doit être désactivé et le fichier RS activé.

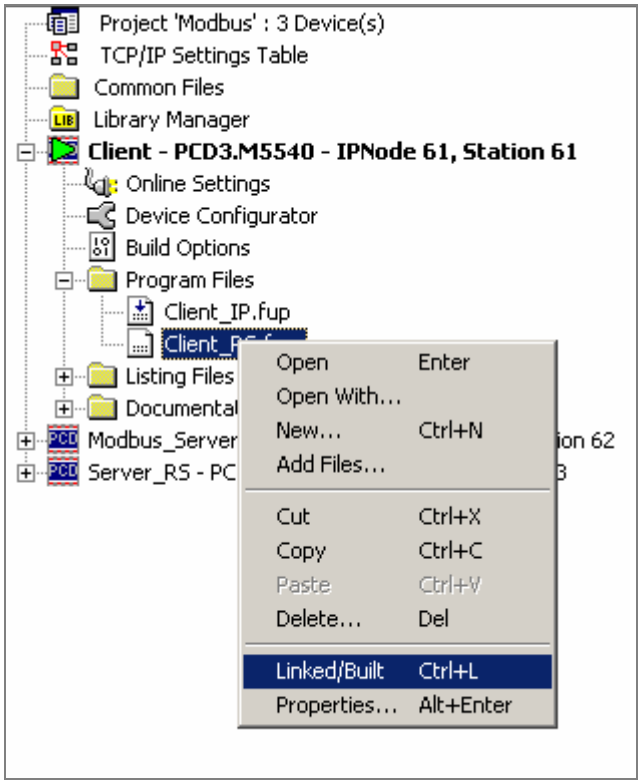
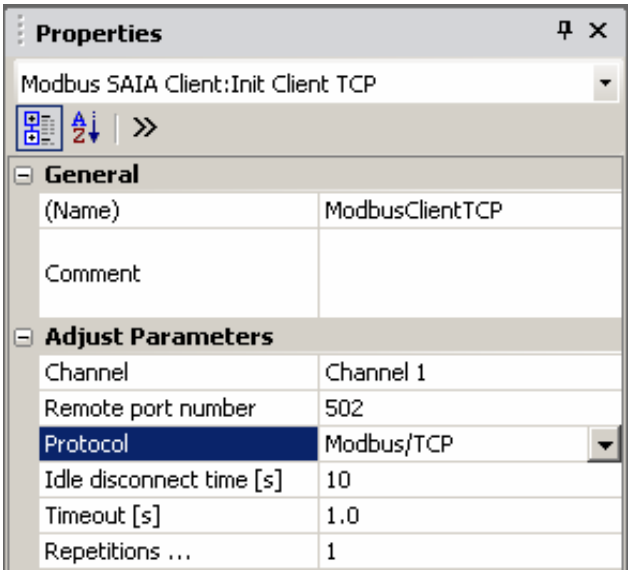


Figure 5.2.1. Etablissement de liens

5.2.1 Communication IP



Dans le fichier fupla Client_IP.fup, vous devrez vérifier et adapter certains réglages en fonction de votre infrastructure.
Une communication Modbus TCP est configurée. Si vous souhaitez une communication Modbus UDP, ce réglage peut être effectué dans la Fbox Init Client TCP. Dans ce cas, il est important que le partenaire de communication présente les mêmes réglages.

Figure 5.2.1.1 Client TCP

Dans la Fbox Def Unit Client, l'adresse IP sera adaptée à celle de l'autre station.

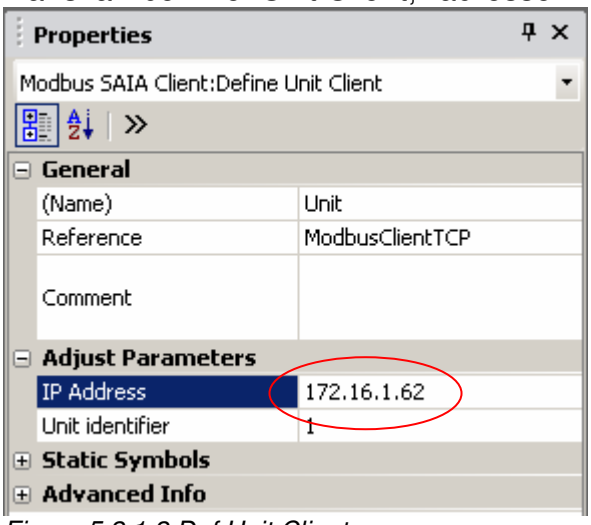


Figure 5.2.1.2 Def Unit Client

5.2.2 Communication RS

Pour une communication sérielle, contrôler et adapter le cas échéant les réglages dans le fichier fupla Client_RS.fup. La communication sérielle s'effectue via le port 2 RS 485 (embarqué) du PCD3. Si un autre port doit être utilisé, cela devra être paramétré dans la Fbox Init Client RS. Lorsque la vitesse de transfert ou le protocole sont modifiés, ce paramétrage devra également être adapté dans la station opposée.

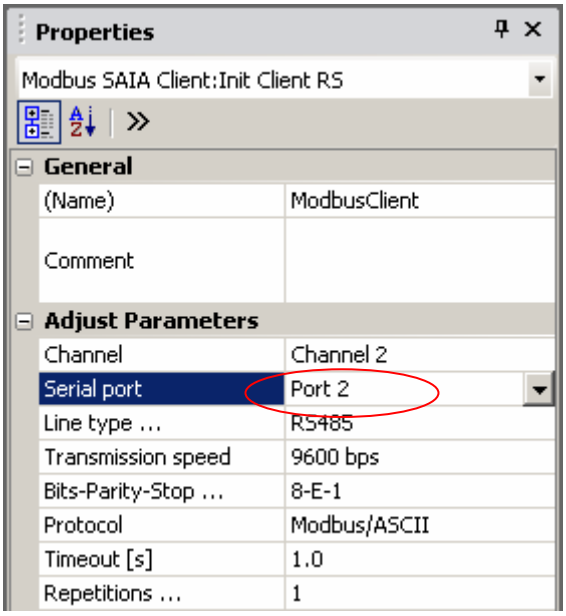


Figure 5.2.2.1 Init Client RS

Sur le Serveur_RS, le port de communication devra aussi être adapté, dans la mesure où la communication n'a pas lieu via le port 2 du PCD3.

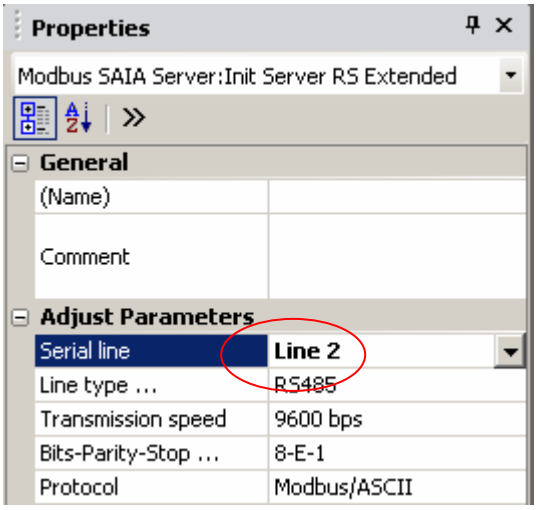


Figure 5.2.2.2 Init Server RS Extended

5.3 Programmer la PCD

Charger le programme dans l'automate

Reste à effectuer la programmation du PCD. A cet effet, le programme doit d'abord être traduit (« Build »). Pour ce faire, vous pouvez utiliser le bouton « Rebuild All».

Après que le « Build » du programme ait été effectuée avec succès, vous pouvez charger le programme dans le PCD à l'aide du bouton « Download Program ». En fonction du réglage de votre PG5, l'automate passe automatiquement en mode RUN après le téléchargement. Si cela n'est pas le cas, paramétrez l'automate sur mode RUN.

6 Programmation du PCD

Ce chapitre contient une description succincte de l'application.

6.1 Client

Cet automate agit en tant que client TCP avec le fichier fupla Client_IP et en tant que client RS (RS 485) avec le fichier fupla Client_RS.

6.1.1 Client_IP

Page 1
Dans la première page de ce programme Fupla a lieu la programmation de la communication Modbus.

La Fbox Init Client TCP définit le canal de communication, le numéro du port distant, le protocole ainsi que le timing.

La Fbox Def Unit Client définit l'adresse IP du serveur et l'identifiant du module. Le Flag Enable Def doit être activé.

La Fbox Read Int permet la lecture des valeurs entières. Le champ Add indique l'adresse vers laquelle les éléments sont copiés (Registre : ReadIP, R 100). # définit le nombre de registres à lire (10 = R 100 à R 109).

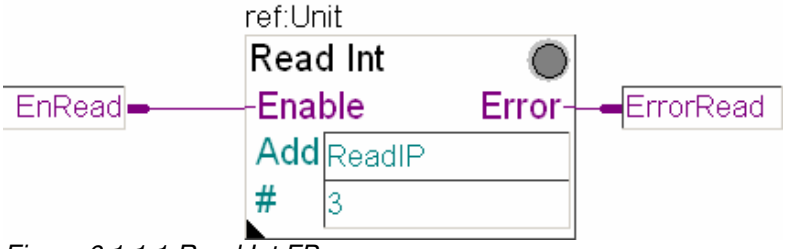


Figure 6.1.1.1 Read Int FBox

On lit un registre de 32 bits. L'adresse de base des Holding Registers est 1. Sur le Server Modbus, les données à transférer (registres PCD 100-102) sont mappées sur les Holding Registers 1 à 6. Comme les Holding Registers n'ont que 16 bits, cela signifie que les registres PCD (32 bits) seront transmis sur 2 Holding Registers.

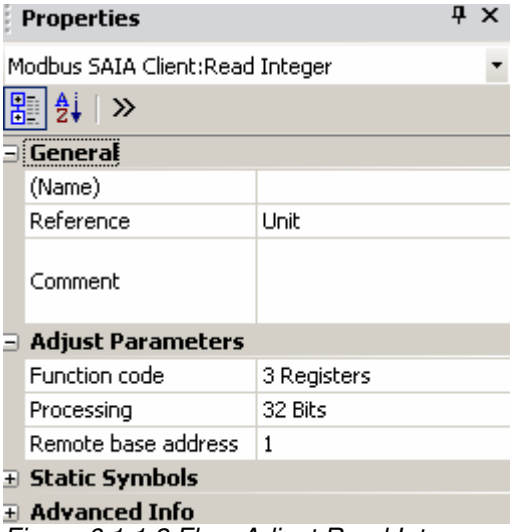


Figure 6.1.1.2 Fbox Adjust Read Int

Le Flag EnRead doit être activé pour que les données soient lues.

Important : si plus d'un registre est surécrit par les données Modbus (3 registres dans ce cas), il est nécessaire de réserver la plage correcte pour les données reçues (tableau de 3 registres PCD).

La Fbox Write Bin permet la écriture des valeurs binaires. 8 éléments sont écrits. Ces Flags à transmettre sont le Flag WriteBin = F 1000 et les Flags suivants (F 1000 à F 1007).
Le Flag EnWrite doit être activé pour démarrer le processus d'écriture. Sur le Modbus, ces Flags sont mappés sur les Coils 1000 à 1007.

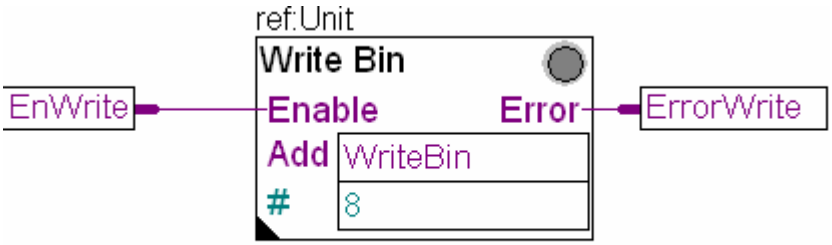


Figure 6.1.1.3 Fbox Write Bin

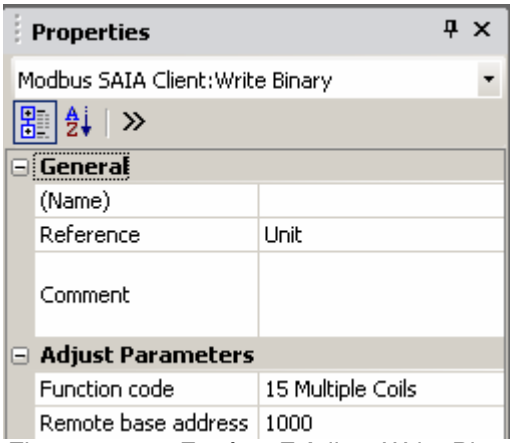


Figure 6.1.1.3 Fenêtre F Adjust Write Bin

Page 2
Les 3 premiers registres reçus via le Modbus fournissent l'heure et la date de l'autre automate. Dans cette page, l'heure est alors transmise à l'horloge hardware de l'automate. Comme le premier registre contient l'heure en heures, minutes et secondes, une division par 100 est nécessaire pour obtenir uniquement des heures et des minutes comme l'exige la Fbox Write Clock.

Page 3
Ici les Flags transmis changent d'état, afin que l'on puisse immédiatement constater du côté opposé que la communication fonctionne.

6.1.2 Client_RS

Page 1

L'initialisation du client RS est réalisée dans cette page. La Fbox Init Client RS permet de définir un canal (virtuel), un port (physique), l'interface physique, les paramètres de communication, le protocole et le timing.

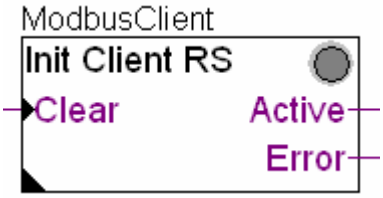


Figure 6.1.2.1 Fbox Init Client RS

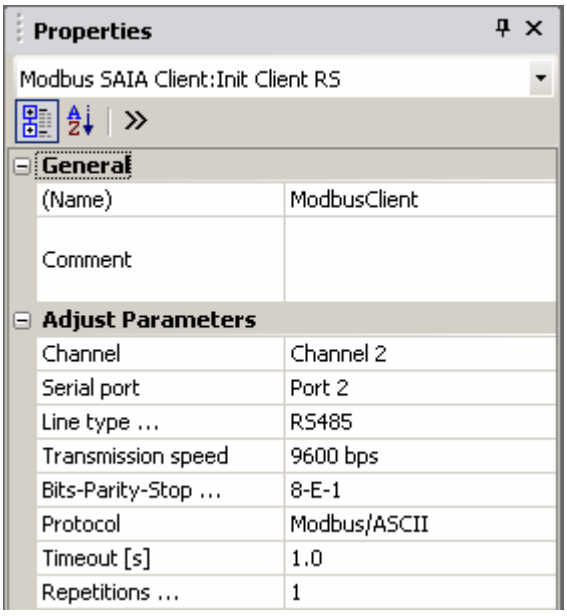


Figure 6.1.2.2 Fbox Adjust Init Client RS

La Fbox Def Unit Client définit l'Unit Identifier UID du partenaire de communication. La saisie d'une adresse IP est inutile dans ce cas. Le champ Adresse IP peut contenir une adresse quelconque.

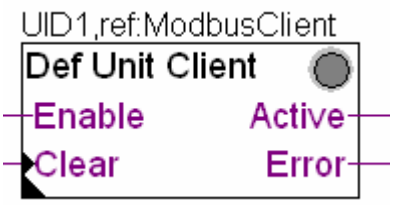


Figure 6.1.2.3 Fbox Def Unit Client

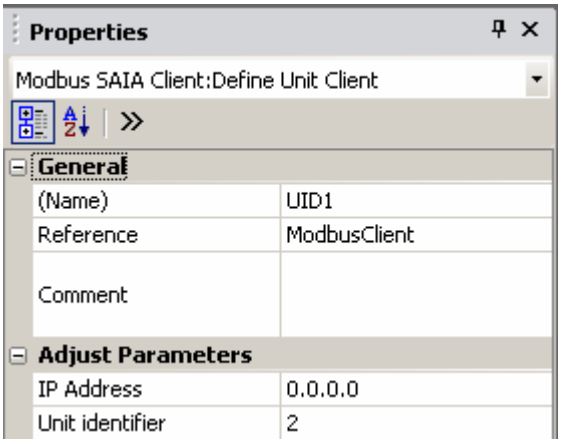


Figure 6.1.2.4 Fbox Def Unit Client

Le Flag EnDefRS doit être activé.

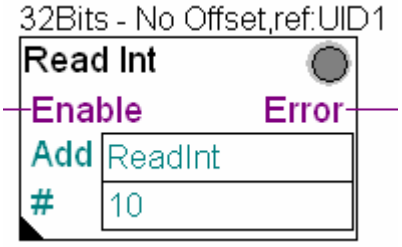


Figure 6.1.2.5 Fbox Read Int

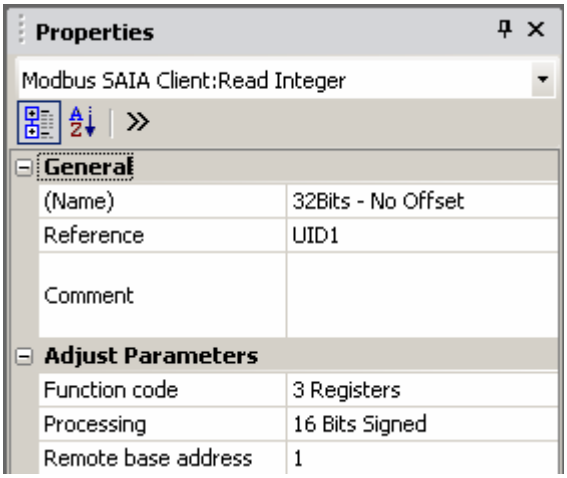


Figure 6.1.2.6 Fbox Adjust Read Int

Des registres 16 Bits signés sont lus à partir de l'adresse de base Modbus 1. Ils sont ensuite mémorisés dans les adresses ReadInt R0 à R 9. Comme le mapping par défaut est activé sur le Server Modbus, les registres PCD 0 à 9 (16 bits signés) sont sur les Holding Registers 1 à 10.

Default Mapping

	Modbus Request			PCD			
Access Type	**MediaType**	**Start Addr.**	**Range**	**MediaType**	**Start Addr.**	**Range**	**AreaType**
ReadWrite	MB_HOLDING_REG_MEDIA	1	10000	PCD_REG_MEDIA	0	10000	MB_AREA_16BITS_SIGNED
ReadWrite	MB_HOLDING_REG_MEDIA	10001	10000	PCD_REG_MEDIA	0	10000	MB_AREA_32BITS
ReadWrite	MB_HOLDING_REG_MEDIA	20001	10000	PCD_REG_MEDIA	0	10000	MB_AREA_32BITS_IEEE
ReadOnly	MB_INPUT_REG_MEDIA	1	10000	PCD_TC_MEDIA	0	10000	MB_AREA_16BITS_SIGNED
ReadOnly	MB_INPUT_REG_MEDIA	10001	10000	PCD_TC_MEDIA	0	10000	MB_AREA_32BITS
ReadWrite	MB_COILS_MEDIA	1	10000	PCD_FLAG_MEDIA	0	10000	MB_AREA_COILS
ReadOnly	MB_DISCR_INPUT_MEDIA	1	10000	PCD_IO_MEDIA	0	10000	MB_AREA_COILS

Figure 6.1.2.7 Mapping

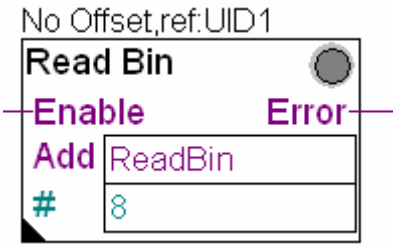


Figure 6.1.2.8 Fbox Read Bin

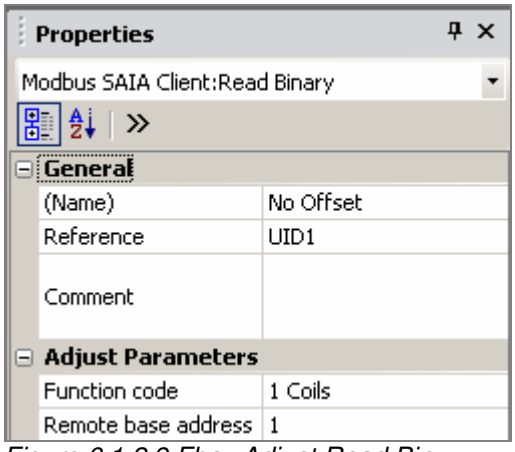


Figure 6.1.2.9 Fbox Adjust Read Bin

8 valeurs binaires sont lues à partir du Coil Modbus 1. Elles sont mémorisées sur les Flags ReadBin F 100 à F 107.

Page 2
Sur les Flags 100 à 107, les valeurs Bin sont transformées en valeurs Int. Si le transfert réussit, il s'en suit une incrémentation.
Les registres lus sont additionnés avec la Fbox d'addition. Ici aussi, la valeur doit changer continuellement lorsque le transfert est actif.

6.1.3 Serveur Modbus (IP)

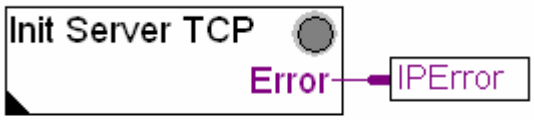


Figure 6.1.3.1 Fbox Init Server TCP

Cette Fbox permet de définir le port et le protocole de la connexion IP.

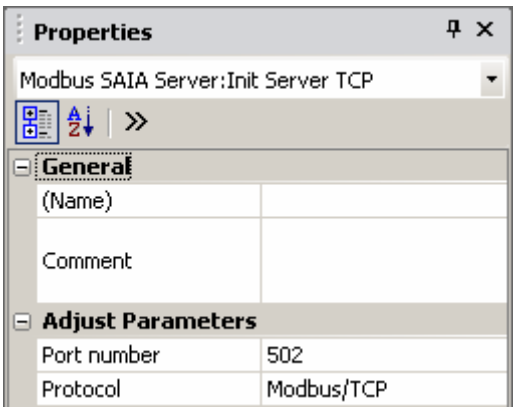


Figure 6.1.3.2 Fbox Adjust Init Server TCP

La Fbox Def Unit Server définit l'Unit Identifier UID. Elle permet également de définir un offset pour ne pas débuter à 0, mais à 1. La permutation de mots peut être activée si nécessaire, de même que les Holes, si cela est souhaité.

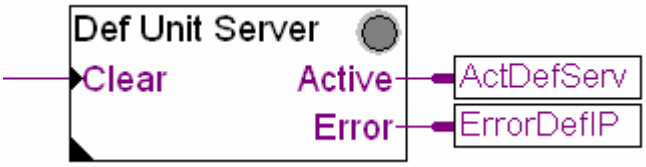


Figure 6.1.3.3 Fbox Def Unit Server TCP

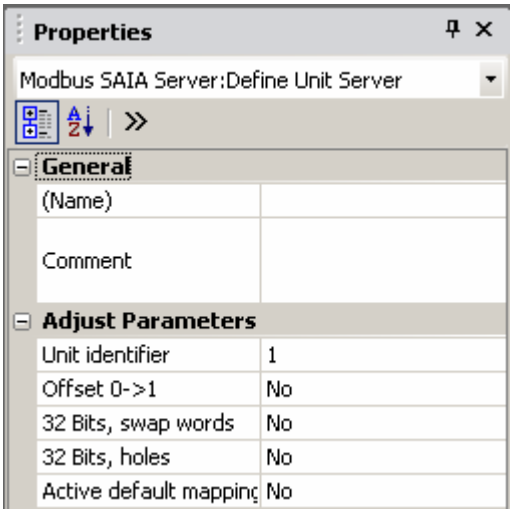


Figure 6.1.3.4 Fbox Adjust Def Unit Server TCP

La Fbox Def Mapp Bin permet de créer un mappage pour la zone binaire.

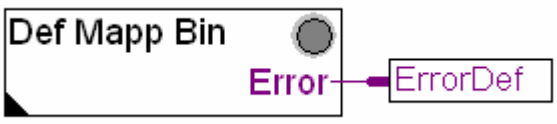


Figure 6.1.3.5 Fbox Def Mapp Bin

Dans ce cas les Coils 1000 à 1007 sont mappés sur les Flags 1000 à 1007.

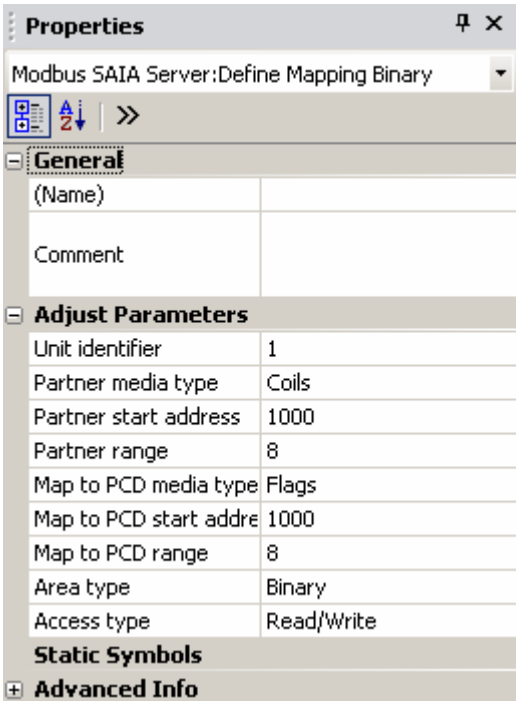


Figure 6.1.3.6 Fbox Adjust Def Mapp Bin

La Fbox Def Mapp Int permet de créer un mappage pour la zone des entiers.

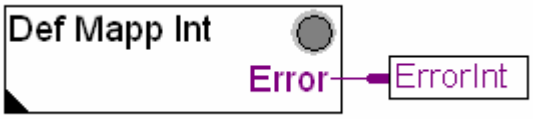


Figure 6.1.3.7 Fbox Def Mapp Int

Les Holding Registers 1 à 6 sont mappés sur les registres PCD 100 à 102. Comme les registres PCD ont 32 bits et que les Holding Registers ont 16 bits, chaque registre PCD sera transféré sur 2 Holding Registers.

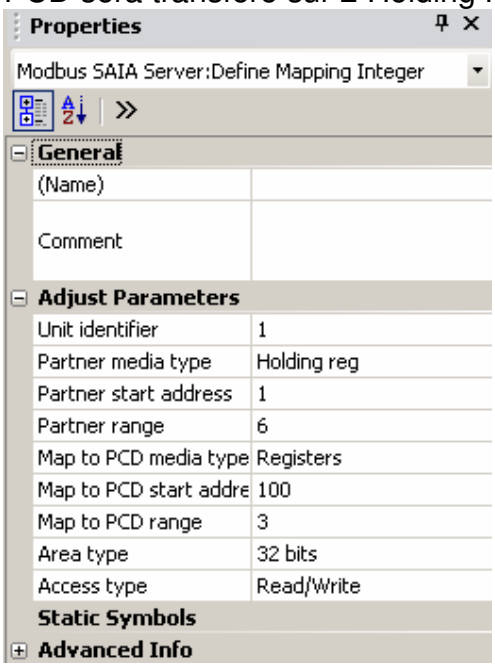


Figure 6.1.3.8 Fbox Adjust Def Mapp Int

Page 2

L'heure et la date du PCD sont lues dans cette page. Elles sont alors transmises au client. De plus, les Flags binaires reçus sont transformés en entiers. Une incrémentation a lieu lors de la communication.

6.1.4 Server_RS

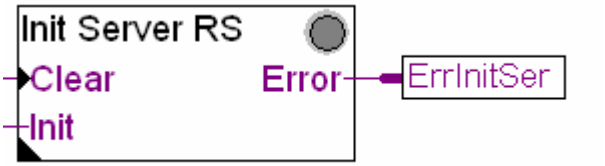


Figure 6.1.4.1 Fbox Init Server RS

Cette Fbox permet l'initialisation de l'interface correspondant pour Modbus.

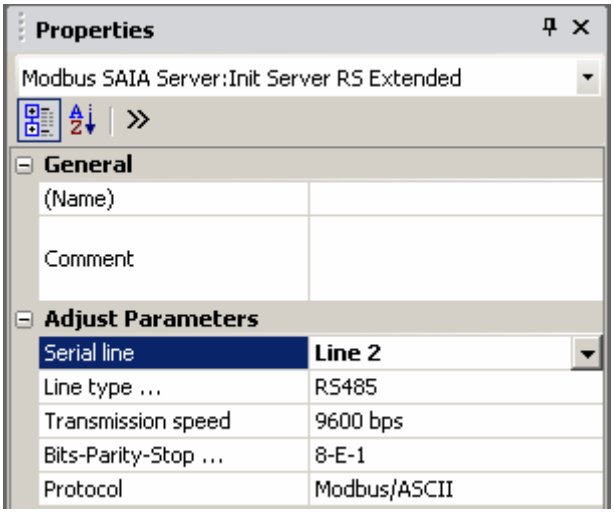


Figure 6.1.4.2 Fbox Adjust Init Server RS

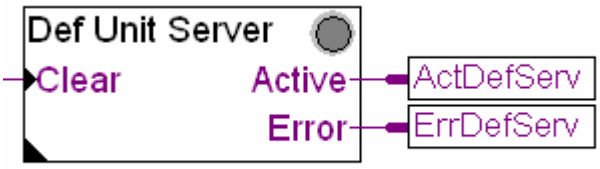


Figure 6.1.4.3 Fbox Def Unit Server

Cette Fbox permet de paramétrer l'Unit Identifier UID et de définir si un offset doit être activé. Elle permet également l'activation de la permutation de mots et les Holes. De plus, le mappage par défaut peut être activé.

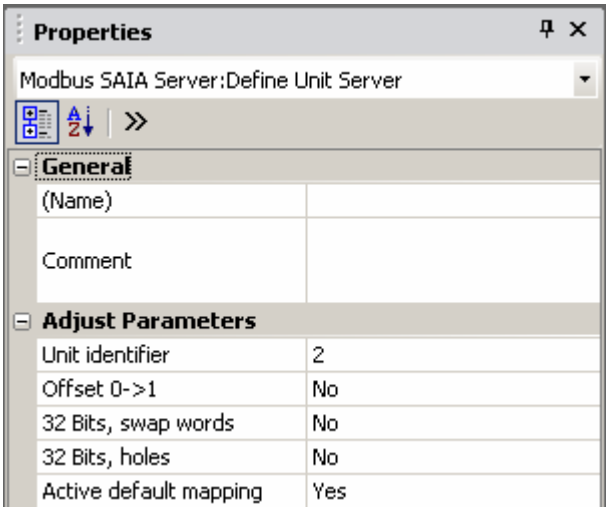


Figure 6.1.4.4 Fbox Adjust Def Unit Server

Les Fbox suivantes activent le clignotement des Flags à transmettre et les changements de valeurs des registres à transmettre.

6.2 Données transmises

6.2.1 IP Ethernet

Les registres 100 à 102 du Serveur IP sont transmis aux Holding Registers 1 à 6. Sur le Client IP, ils sont alors copiés sur les registres 100 à 102.
Les Flags 1000 à 1007 du Client sont écrits sur les Coils 1000 à 1007. Sur le Server, ces Coils 1000 à 1007 sont mappés sur les Flags 1000 à 1007.

6.2.2 Sériel RS 485

Les Flags 0 à 7 sont transmis du Serveur_RS aux Coils Modbus 1 à 8. Ceux-ci sont alors écrits dans le Client_RS sur les Flags 100 à 107.

Les registres 0 à 9 (16 bits) sont transmis aux Holding Registers Modbus 1 à 10 et écrits dans le client sur les registres 0 à 9.

7 Recherche d'erreurs

Symptôme	Cause possible	Solution
Le mappage par défaut n'est pas lu, bien que dans la Fbox "Default Mapping" soit réglée sur "Yes".	Dès qu'un mappage est configuré pour une UID, le mappage par défaut est désactivé.	Supprimer sur le PCD tous les mappages pour cette UID.
Les données ne sont pas reçues, ne sont pas transmises, bien que la Fbox soit verte.	Les données sont éventuellement transmises à une autre adresse.	Vérifier le mappage.
La Fbox est rouge, la communication ne fonctionne pas ou seulement partiellement.	Les paramètres de communication ne correspondent pas à ceux de la station opposée. Le port est déjà configuré ailleurs (Hardware Settings). Le câblage n'est pas correct.	Vérifier les paramètres de communication, la configuration et le câblage.
Un projet réalisé préalablement avec la bibliothèque Engiby ne fonctionne pas avec la bibliothèque Modbus Saia.	Le mappage ne correspond pas. Les désignations différentes peuvent entraîner des confusions.	Vérifier selon le chapitre 3.6.1 si le mappage a été repris correctement.
La Fbox "Define Unit Server" présente une erreur du type "Range Error".	La plage n'est pas correctement définie (par ex. longueur erronée) ou seule un "demi" registre PCD est lu pour un mappage de 32 bits (par ex. uniquement HR 0 au lieu de 0 et 1).	Vérifier la plage et contrôler si les registres 32 bits sont lus correctement.

8 Références

Thème	Document	N°
Modbus	Manuel Modbus	26/866
Divers	Saia® FAQ Manager www.sbc-support.ch/faq	-