MODBUS Protocol

Introduction

Ce protocole définit une structure de message que les contrôleurs reconnaîtront et utiliseront, quel que soit le type de réseau sur lequel ils communiquent.

Il décrit le processus qu'un contrôleur utilise pour demander l'accès à un autre appareil, la manière dont il répondra aux demandes des autres appareils et la manière dont les erreurs seront détectées et signalées.

Le protocole Modbus est mis en œuvre sur les contrôleurs Modicon.

Technique Master-Slave

Un seul maître peut initier la transaction (requête). Les autres (esclaves) répondent en fournissant les données demandées au maître, ou en effectuant l'action demandée dans la requête.

Exemples de maîtres : raspberry pi 3.

Exemples d'esclaves: panneaux photovoltaique.

Le maître peut s'adresser à des esclaves individuels, ou peut lancer un message de diffusion à tous les esclaves qui leur sont adressés individuellement.

Mode de transmission en série

Deux modes de transmission en série :

- ASCII

- RTU

Le mode et les paramètres de série doivent être les mêmes pour tous les appareils d'un réseau Modbus, puisque ces deux modes sont incompatibles.

ASCII

Chaque 8 bits sont envoyés sous forme d'un caractères ASCII.

Avantage -> Plus d'intervalles de temps entre les caractères sans provoquer d'erreur.

ASCII Message Frame

Start 1 char (:)	Slave Address 2 char	Function Code 2 char	Data N char	LRC check 2 char	End 2 char (CRLF)

RTU

Chaque 8 bits contiennent deux caractères hexadécimaux de 4 bits.

Avantage -> Meilleur débit de données.

RS-232

Si vous n'avez besoin de connecter que 2 appareils ensemble, et que la distance entre les 2 appareils est inférieure à 15 mètres.

RS-422

Pour connecter plus de 2 appareils sur la même ligne, et avoir une distance supérieure à 15 mètres.

RS-485

Pour une communication maître avec plusieurs appareils esclaves. Il peut prendre en charge plus de 32 nœuds sur une portée allant jusqu'à 1200 mètres, sans répéteur.

Baud-rate

La vitesse à laquelle les messages Modbus sont envoyés est appelée baud-rate ou bits par seconde. Tous les appareils d'un réseau RTU doivent utiliser le même baud-rate.

9600-19200 : vitesse typique

300-100000+ : vitesse possible

RTU Network

Les appareils doivent être reliés en guirlande.

!! Ils ne peuvent pas être connectés selon une topologie en étoile.

RTU Message Format

Start T1-T2-T3-T4	Slave Address 8 bits	Function Code 8 bits	Data N x 8 bits	CRC check 16 bits	End T1-T2-T3-T4

N.B:

- La plage d'adresses de l'esclave : 1-247
- L'adresse 0 est pour la diffusion.
- Le maître envoie le code de fonction 0000 0011
  S'il y a une erreur, l'esclave renvoie 1000 0011
  S'il n'y a pas d'erreur, l'esclave fait écho à 0000 0011
- Les caractères sont transmis en série, du moins significatif au plus significatif.

MODBUS TCP/IP

Le Modbus TCP/IP utilise le terme de client-serveur au lieu de maître-esclave. Le réseau TCP/IP se compose du client connecté à un commutateur, ou à une série de commutateurs, auquel tous les serveurs du réseau sont également connectés.

Le réseau Modbus TCP/IP utilise des adresses de protocole Internet (IP) et nécessite un masque de sous-réseau. La passerelle par défaut est facultative.

Technique Client-Serveur

Les clients sont les maitres et le serveur est l'esclave.

Ce sont les clients qui doivent lire et écrire dans le serveur Modbus.

Chaque client doit se connecter au serveur en protocole TCP (adresse IP du serveur, port 502).

On peut avoir plusieurs maitres (clients).

Les clients ne se connectent pas à d'autres clients.

Les serveurs ne font pas de demandes.

Modèle Client-Serveur

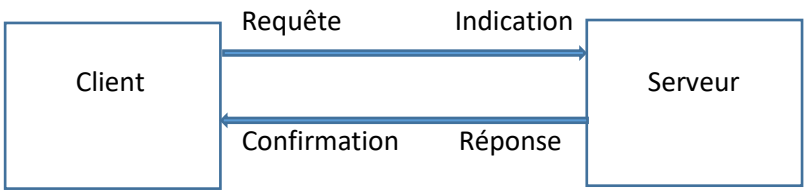
Basé sur quatre types de messages :

- Requête Modbus

- Confirmation de Modbus

- Indication Modbus

- Réponse Modbus



Une requête modbus : c'est le message envoyé sur le réseau par le client pour initier une transaction.

Une indication Modbus : c'est le message de requête reçu du côté du serveur.

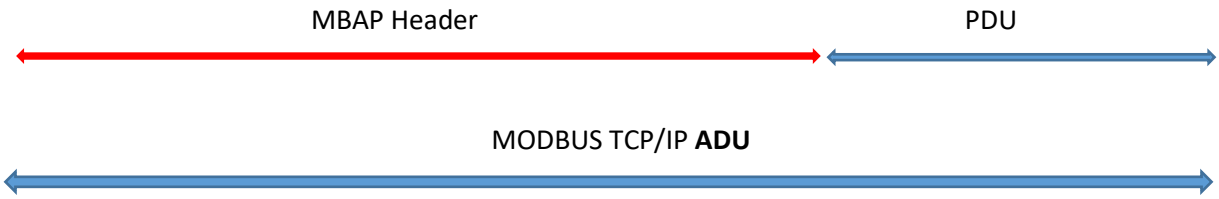
Une réponse Modbus : c'est le message de réponse envoyé par le serveur.

Une confirmation Modbus : c'est le message de réponse reçu du côté client.

TCP/IP message format

Pour initier une transaction, l'appareil construit **l'Application Data Unit** (ADU).

Transaction ID	Protocol ID	Length	Unit ID	Function Code	Data



Transaction ID: Il est utilisé pour le couplage des transactions. Le serveur Modbus copie dans la réponse l'identificateur de transaction de la demande.

Protocol ID: Il est utilisé pour le multiplexage intra-système. Le protocole Modbus est identifié par la valeur 0x0000.

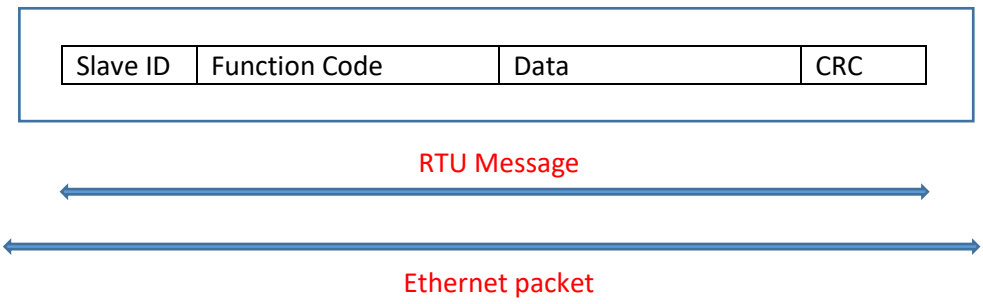
Length: la longueur est un nombre d'octets des champs suivants, y compris Unit ID et Data.

Unit ID: ce champ est utilisé à des fins de routage intra-système. Il est généralement utilisé pour communiquer avec un Modbus + ou un esclave de ligne série Modbus par l'intermédiaire d'une passerelle, d'un routeur ou d'un bridge entre le réseau Ethernet TCP/IP et une ligne série Modbus.

Ce champ est défini par le client Modbus dans la requête et doit être renvoyé avec la même valeur dans la réponse par le serveur.

Message RTU Encapsulé dans un packet TCP/IP

Les messages série Modbus peuvent également être envoyés sous forme de messages RTU ordinaires encapsulés dans un paquet Ethernet TCP/IP. Les messages encapsulés peuvent utiliser n'importe quel port.



!! Notez que les MBAP et RTU encapsulés ne sont pas compatibles.

Les appareils doivent être configurés pour utiliser l'un ou l'autre.

Confirmation Modbus

Dans le header MBAP :

- Si l'identificateur de transaction ne fait pas référence à une transaction Modbus en cours, la réponse doit être rejetée.

- Sinon, la réponse doit être analysée.

- Vérifier l'identificateur de protocole (=0x0000).
- Si le serveur est directement connecté au réseau TCP/IP, l'identificateur d'unité (0x00FF) est insignifiant et doit être rejeté.
- Si le serveur est connecté sur un sous-réseau de lignes série et que la réponse provient d'un bridge, d'un routeur ou d'une passerelle, l'Unit Identifier (!= 0x00FF) identifie le serveur.

Dans le packet PDU :

1. Function Code serveur = function code client ET format correcte
   ➔ Réponse positive
2. Function Code serveur = exception code
   ➔ Réponse positive
3. Function code serveur # function code client OU format incorrecte
   ➔ Réponse négative

N.B : **NumberMaxOfServerTransaction** : paramètre qui définit le nombre de requêtes Modbus simultanées que le serveur peut accepter (1<nb>16).

MODBUS storage tables

Coils (1-10000): 1bit Read-Write 1 bit
Discrete Input (10001-20000): 1 bit Read 1 bit
Input Registers (30001-40000): 16 bits Read 16 bits
Output Registers (40001-50000): 16 bits Read-Write 16 bits

Modbus Function Codes

Les codes de fonction Modbus sont des codes numériques qui indiquent à l'esclave à quelle table il doit accéder et s'il doit lire ou écrire dans cette table. Chaque code de fonction se rapporte à une plage d'adresses de table de données spécifique.

01	Read coil status
02	Read input status
03	Read holding registers
04	Read input registers
05	Write single coil status
06	Write single register
0A	Multiple coil write
0B	Multiple register write

Modbus Exceptions

Dans le cas d'une réponse exception :

Function code serveur = function code client + 80H.

Quelques examples d'exceptions :

01	Illegal code
02	Illegal address
03	Illegal data value
04	Server failure

05	Acknowledge
06	Server busy
0A	Gateway problem
0B	Gateway problem