

1 – Interrogation d'une entrée numérique

Programmation M221

L'automate utilisé est le M221, la documentation est donnée en annexe.

Les fonctions Modbus supportées par l'automate M221 sont les suivantes :

What are the supported Modbus Function Codes for a M221 controller?

This table lists the function codes supported by both serial Modbus and Modbus TCP and their effect on controller memory variables:

Supported Modbus Function Code	Supported Sub-Function Code	Description
1 (0x01)	-	Read multiple internal bits %M
2 (0x02)	-	Read multiple internal bits %M
3 (0x03)	-	Read multiple internal registers %MW
4 (0x04)	-	Read multiple internal registers %MW
5 (0x05)	-	Force single internal bit %M
6 (0x06)	-	Write single internal register %MW
8 (0x08)	0 (0x00), 10 (0x0A)...18 (0x12)	Diagnostics
15 (0x0F)	-	Write multiple internal bits %M
16 (0x10)	-	Write multiple internal registers %MW
23 (0x17)	-	Read/write multiple internal registers %MW
43 (0x2B)	14 (0x0E)	Read device identification (regular service)

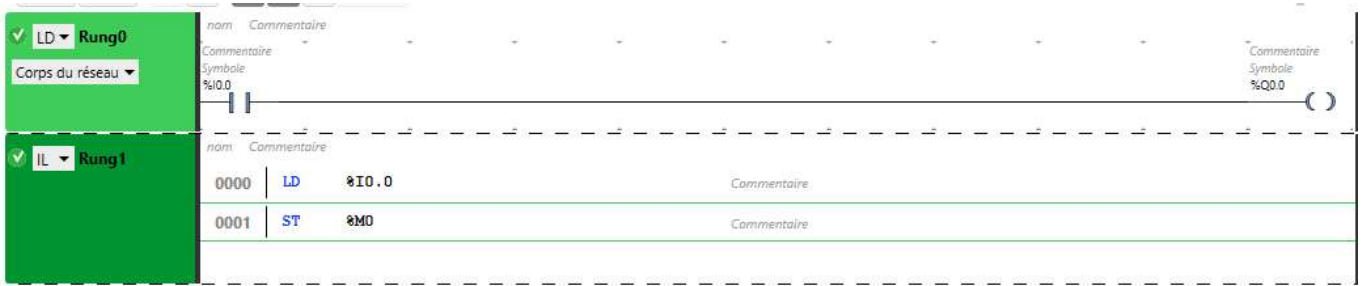
Note: For function code 5 and 6 you must use the EXCH function. These two function codes are not available with the

Donc il faudra utiliser les mémoires %M pour les bits et %MW pour les mots binaires (de 16 bits).

On fera les tests en mode simulation, l'adresse IP à saisir pour l'automate sera donc celle du PC.

- ⇒ Lancer le logiciel EcoStructure
- ⇒ Créer un nouveau projet
- ⇒ Dans le menu « Propriétés », désactiver les protections en lecture et en écriture.
- ⇒ Dans le menu « Configuration », Choisir l'automate M221CE24R
- ⇒ Dans le menu « Configuration/ETH1 », entrer l'adresse fixe et le masque de sous réseau. Valider les protocoles de programmation, EtherNet/IP, Serveur Modbus et recherche automatique.

⇒ Dans le menu « Programmation », réaliser le programme suivant :



⇒ Dans le menu « Mise en service », lancer le simulateur et démarrer le contrôleur.

⇒ Dans le menu « Programmation », visualiser le fonctionnement en simulation.

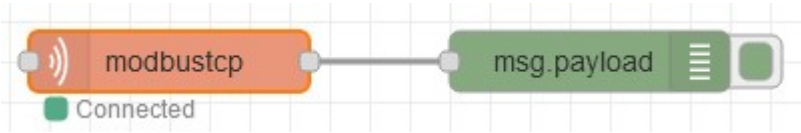


Interrogation de l'automate par NODERED

⇒ Vérifier la présence des nodes Modbus dans la palette



⇒ Réaliser le flux suivant :



Edit modbustcp node

Delete Cancel Done

Properties

Name:

Topic:

FC:

Address:

Quantity:

Poll Rate:

Server:

IEEE-754 # Type:

Endian:

Edit modbustcp node > Edit modbustcp-server node

Delete Cancel Update

Properties

Name:

Host:

Port:

Unit Id:

Reconnect Interval (s):

* mettre la bonne adresse IP

⇒ Visualiser l'évolution de la mémoire %M0 en agissant sur I0.0 (en simulation)

debug

20/10/2022 18:29:10 node: 9fdb2581.717ea
msg.payload : array[8]
▶ [false, false, false, false, false, false, false, false]

20/10/2022 18:29:15 node: 9fdb2581.717ea
msg.payload : array[8]
▶ [true, false, false, false, false, false, false, false]

Remarque : La fonction 0x02 retourne un octet complet, et non un bit uniquement.

Request

Function code	1 Byte	0x02
Starting Address	2 Bytes	0x0000 to 0xFFFF
Quantity of Inputs	2 Bytes	1 to 2000 (0x7D0)

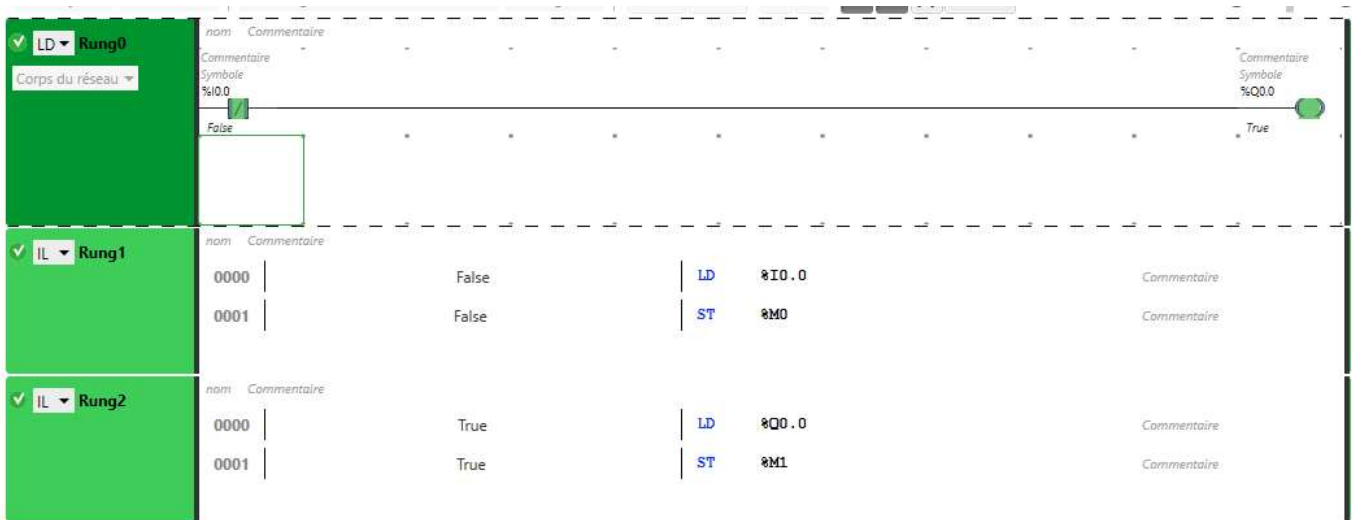
Response

Function code	1 Byte	0x02
Byte count	1 Byte	N*
Input Status	N* x 1 Byte	

*N = Quantity of Inputs / 8 if the remainder is different of 0 ⇒ N = N+1

2 – Interrogation d'une sortie numérique

⇒ Modifier la programmation de la manière suivante :



Edit modbustcp node

Delete Cancel Done

Properties

Name:

Topic:

FC:

Address:

Quantity:

Poll Rate:

Server:

IEEE-754 # Type:

Endian:

Edit modbustcp node > Edit modbustcp-server node

Delete Cancel Update

Properties

Name:

Host:

Port:

Unit Id:

Reconnect Interval (s):

⇒ Visualiser l'évolution de I0.0 et Q0.0 dans la fenêtre DEBUG de NODERED

3 – Interrogation d'une entrée analogique

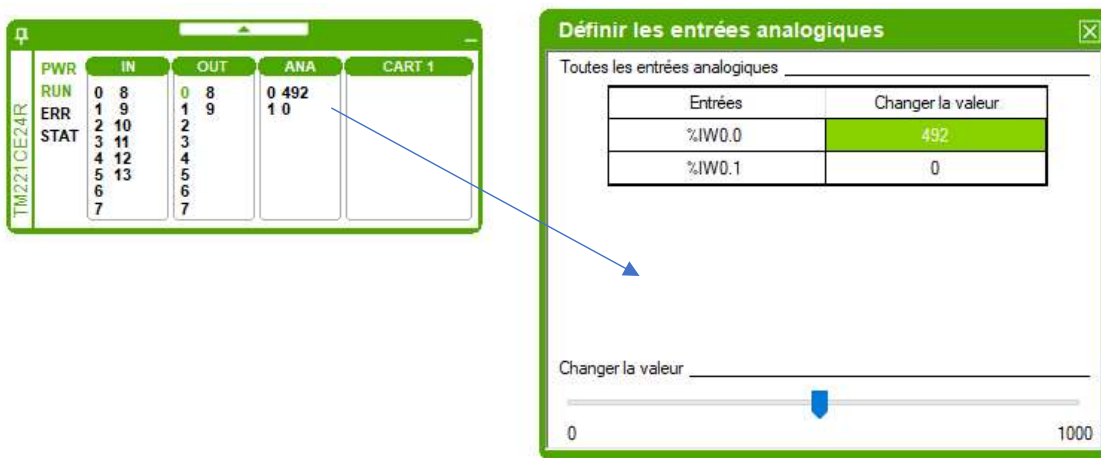
⇒ A partir du menu Programmation/Outils/Entrées analogiques, noter les adresses des voies analogiques 0 et 1.

⇒ Ajouter le réseau suivant à la programmation :

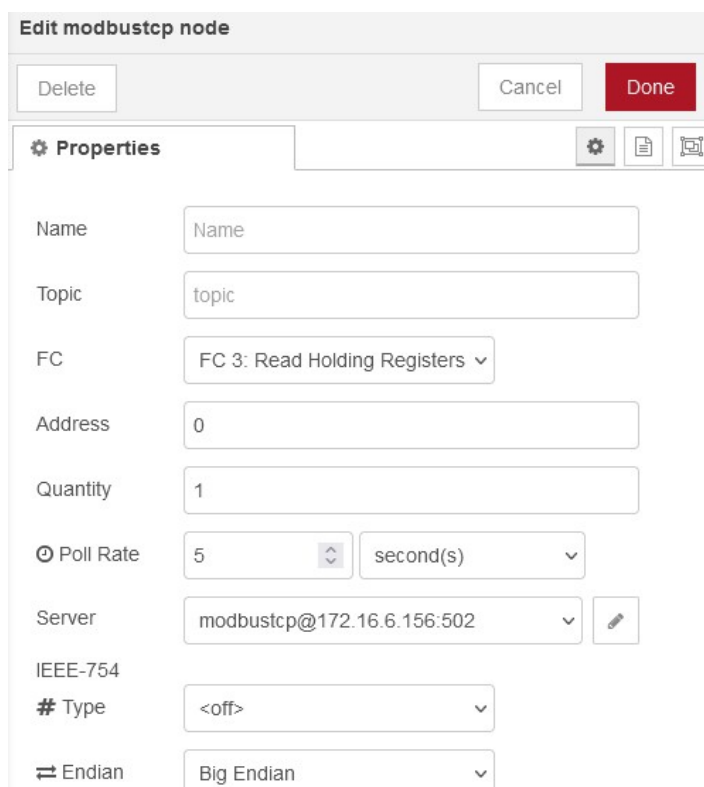


⇒ Lancer le simulateur et démarrer le contrôleur

⇒ Double cliquer sur la voie 0 de ANA et modifier la valeur de %IWO.0 avec le curseur



⇒ Sous NODERED, modifier les paramètres du NODE Modbus pour lire un registre à partir de l'adresse 0.



⇒ Visualiser l'évolution de la mémoire %MW0 dans la fenêtre DEBUG de NODERED, et agir sur le curseur dans EcoStructure.

The image shows two side-by-side screenshots. The left screenshot is a window titled "Définir les entrées analogiques" (Define analog inputs). It contains a table with two columns: "Entrées" (Inputs) and "Changer la valeur" (Change value). The table has two rows: the first row shows "%IW0.0" with a value of "604", and the second row shows "%IW0.1" with a value of "0". Below the table is a slider control labeled "Changer la valeur" with a range from 0 to 1000 and a blue handle positioned at approximately 604.

The right screenshot is the Node-RED debug console. It shows two messages. The first message, received at 21/10/2022 09:58:54, has a payload of an array with one element: `array[1]` containing the value `604`. The second message, received at 21/10/2022 09:58:59, has a payload of an array with one element: `[604]`.

Pour enregistrer plus simplement les données dans une base données, il est préférable d'avoir un format JSON.

⇒ Insérer la fonction ci-dessous et visualiser la transformation d'un tableau (avec un seul élément ici) en format JSON.

The image shows a Node-RED flow and the configuration for a function node. The flow consists of a "modbus tcp" node (labeled "Connected") connected to a "function" node, which is then connected to a "msg.payload" output node. A blue arrow points from the "function" node in the flow to the "Edit function node" configuration window.

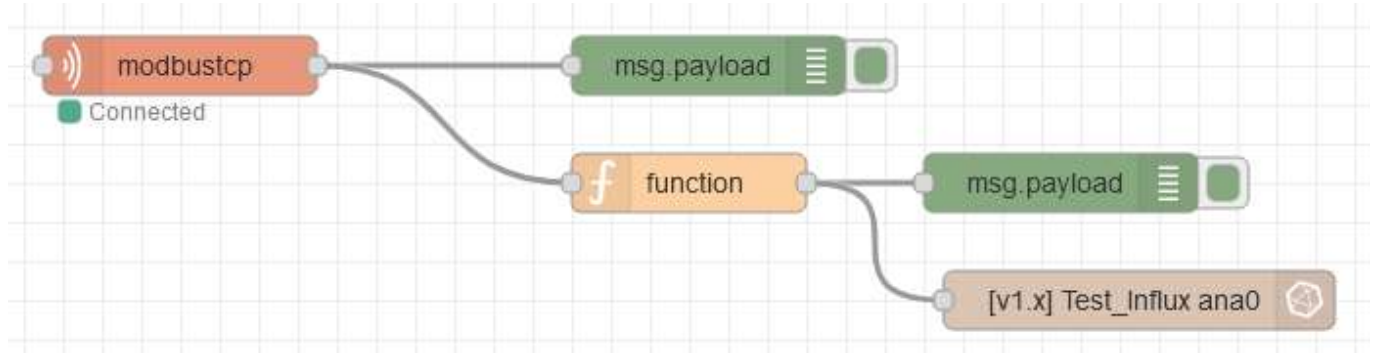
The "Edit function node" window shows the following configuration:

- Name:** Name
- On Message:** Selected
- Code:**

```
1 msg.payload={"ANA0":msg.payload[0]};
2 return msg;
```

The debug console on the right shows the transformation of the payload. The first message (at 10:16:17) shows the original payload as an array: `array[1]` containing `604`, labeled "Tableau". The second message (at 10:16:17) shows the transformed payload as a JSON object: `{ ANA0: 604 }`, labeled "JSON".

⇒ Ajouter l'enregistreur dans la base de données TPTSMI (créée dans un TP précédent) sur Raspberry, dans une table (measurement) appelée ana0. Enregistrement toutes les 5s.



⇒ Visualiser les données enregistrées et agir sur le curseur dans la simulation d'EcoStructure.

```
> select * from ana0
name: ana0
time                ANAO
----                -
1666342871271867469 604
1666342876262084121 604
1666342881246864894 604
1666342886262800702 604
1666342891278327756 772
```

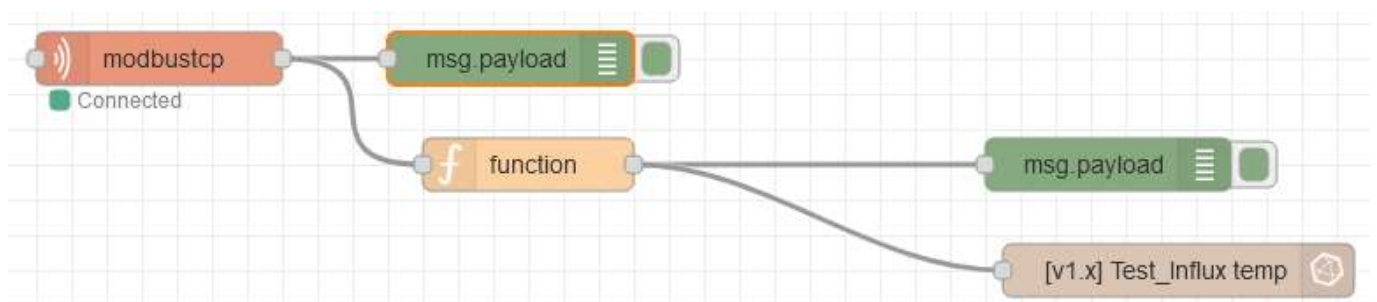
4 – Exercice

Un convertisseur de température PT100 (référence KOS819B, vu au TP1) est réglé pour fournir une tension de 0-10V lorsque la température varie de 0°C à 250°C.

Le convertisseur de l'automate M221 a une résolution de 10mV.

Résolution d'entrée analogique	10 bits
Valeur du bit de poids faible	10 mV
Temps de conversion	1 ms par voie

⇒ Réaliser le flux sous NODERED pour enregistrer dans la base de données TPTSMI, dans une table (measurement) appelée temp, la valeur de la température.



Edit function node

Delete

Properties

Name:

Setup On Start **On Message**

```

1 var data=msg.payload[0];
2 var temp=data*250/1000;
3 msg.payload={"ANA0":temp};
4 return msg;

```

```

21/10/2022 11:32:53 node: 9fdb2581.717ea
msg.payload : array[1]
  [ 500 ]

21/10/2022 11:32:54 node: a87ff46.1817988
msg.payload : Object
  { ANA0: 125 }

```

```

> select * from temp
name: temp
time          ANA0
----          -
1666344641614697028 193
1666344646618796153 250
1666344651672022578 250
1666344656643888147 250
1666344661656415040 0
1666344666656805123 114
1666344671686832382 123.75
1666344676673169554 125
1666344681781859363 125
1666344686688484470 125
1666344691686420916 125
1666344696753097334 125

```

⇒ Créer sous GRAFANA un tableau de bord pour afficher l'évolution de la température

