Aplicație pentru monitorizarea resurselor SO (Client Modbus TCP)

Documentație

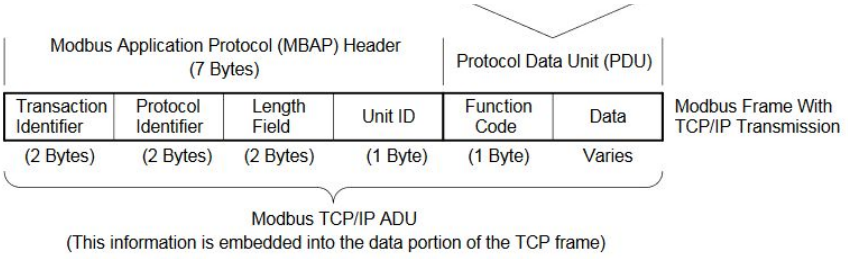
Ioana Ichim

Gafencu Gabriel

Dispozitivele Modbus comunică utilizând o tehnică master-slave (client-server) în care un singur dispozitiv (master / client) poate iniția tranzacții (numite interogări). Celelalte dispozitive (slaves / servere) răspund furnizând datele solicitate masterului sau luând măsurile solicitate în interogare. Un slave este orice dispozitiv periferic care procesează informații și își trimite ieșirea către master folosind Modbus.

Masterul (sau Clientul în Modbus TCP) stabilește o conexiune cu Slave (sau Server). Serverul așteaptă o conexiune primită de la Client. Odată stabilită o conexiune, serverul răspunde la cererile de la client până când clientul închide conexiunea.

Protocolul MODBUS definește o unitate de date de protocol (PDU) simplă.Maparea protocolului MODBUS poate introduce câteva câmpuri suplimentare pe Application Data Unit (ADU).



Modelul de date Modbus are o structură simplă care cu patru tipuri de date de bază:

Object type	Access	Size	Address Space
Coil	Read-write	1 bit	00001 - 09999
Discrete input	Read-only	1 bit	10001 - 19999
Input register	Read-only	16 bits	30001 - 39999
Holding register	Read-write	16 bits	40001 - 49999

Programul va folosi biblioteca psutil pentru a obține informații legate de starea sistemului de operare, precum procentul de utilizare a procesorului, memoria RAM utilizată, procentul de memorie valabil.

În discrete input vor fi mapate care dintre resursele monitorizate este transmisă (i.e CPU usage 10001, RAM usage 10002, Disk space available 10003).

Apoi, procentul utilizat de CPU corespunde primilor 2 regiștrii din Input Registers. RAM-ul utilizat va fi mapat pe primii 2 regiștrii din Holding Registers, iar spațiul valabil de pe disk pe următorii 4.

Protocolul Modbus definește mai multe coduri de funcții pentru accesarea regiștrilor Modbus. Există patru blocuri de date diferite definite de Modbus, iar adresele sau numerele de registru din fiecare dintre acestea se suprapun. Prin urmare, o definiție completă a locului unde se găsește o bucată de date necesită atât adresa (sau numărul de registru), cât și codul funcției (sau tipul de registru).

Function Code	Register Type
1	Read Coil
2	Read Discrete Input
3	Read Holding Registers
4	Read Input Registers
5	Write Single Coil
6	Write Single Holding Register
15	Write Multiple Coils
16	Write Multiple Holding Registers

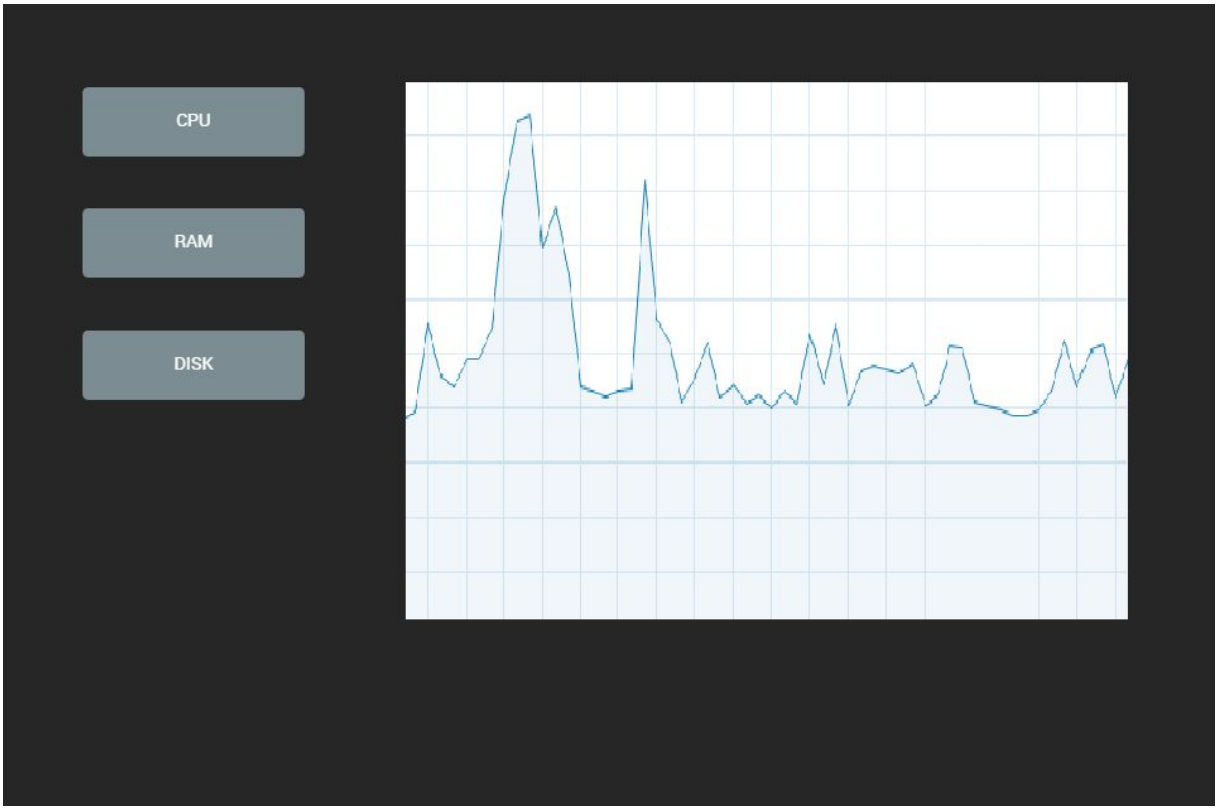
Când un Slave Modbus recunoaște un pachet, dar determină că există o eroare în cerere, acesta va returna un răspuns de cod de excepție în locul unui răspuns de date. Răspunsul la excepție constă din adresa slave sau numărul unității, o copie a codului funcției cu setul de biți puși pe 1 și un cod de excepție. Codurile de excepție pot fi:

1	Illegal Function	The function code received in the query is not recognized by the slave or is not allowed by the slave.
2	Illegal Data Address	The data address (register number) received in the query is not an allowed address for the slave, i.e., the register does not exist. If multiple registers were requested, at least one was not permitted.
3	Illegal Data Value	The value contained in the query's data field is not acceptable to the slave.
4	Slave Device Failure	An unrecoverable error occurred while the slave was attempting to perform the requested action
6	Slave Device Busy	The slave is engaged in processing a long-duration command. The master should try again later.

Pentru mecanismul de autentificare biții din coils de la adresele 7, 77, 777, 7777 vor fi trecuți pe 1. Serverul va verifica ca prima cerere să respecte această convenție. Funcția de acces folosită pentru scrierea în coils va fi funcția 15.

User Interface

*work in



progress

References

1. BusWorks® 900EN Series 10/100M Industrial Ethernet I/O Modules w/ Modbus Technical Reference – Modbus TCP/IP, https://www.prosoft-technology.com/kb/assets/intro_modbustcp.pdf

https://www.automation.com/en-us/articles/2012-1/introduction-to-modbus