Exercices réseaux partie N°2

1.1 Trame modbus sur port série

Soit la classe c++ "PortSerie" permettant la communication avec la couche physique par le port série d'un PC, elle comporte entre autres les méthodes:

- Le constructeur:

```
PortSerie::PortSerie(int bauds=9600, int nbits=8, bool bin=true, bool parity=false);
```

- Les fonctions membres:

```
bool PortSerie::OpenCOM(int nId);
bool PortSerie::CloseCOM(int nId);
```

nId représentant le numéro du port,

```
bool PortSerie::ReadCOM(void* buffer, int nBytesToRead, int* pBytesRead);
bool PortSerie::WriteCOM(void* buffer, int nBytesToWrite, int* pBytesWritten);
```

Ces méthodes permettent d'envoyer ou de reçevoir des octets (unsigned char) stockés dans un tableau (void* buffer qui est équivalent à void buffer[]). Les entiers permettant le contrôle de la taille du tableau

On dispose des fonctions annexes:

```
int int_vf_cp2(double value, int dp);
```

Cette fonction permet de convertir un nombre en virgule flottante en nombre 16 bits à virgule fixe stocké dans un int

```
unsigned int CRC16(unsigned char trame[], unsigned int longueur);
```

Cette fonction renvoie le code de contrôle d'un tableau dont la taille est donnée. Le CRC (2 octets) est ajouté automatiquement à la fin de la table trame[] .

Soit une autre classe C++ "ModbusRTU" utilisant la classe "PortSerie" pour communiquer avec la couche physique, elle comporte entre autres les méthodes:

```
ModbusRTU::ModbusRTU(PortSerie * port_serie);  // constructeur
```

et la méthode d'écriture:

```
bool ModbusRTU::WriteWord(unsigned char slave, unsigned int adr, double value, unsigned int
dp);
```

Elle permet d'écrire dans l'esclave (code modbus 6) d'adresse slave (8bits) à l'indexe adr (16 bits) un nombre en virgule flottante (value) sous forme d'un nombre en virgule fixe (16bits) dont la position de la virgule est spécifiée par dp

- Ecrire une ligne d'instructions permettant de copier les 8 bits de poids fort d'un mot de 16 bits non signé (unsigned int) dans un octet (unsigned char), plusieurs solutions
- Ecrire une ligne équivalente pour les 8 bits de poids faible.
- Compléter cette fonction:

```
bool ModbusRTU::WriteWord(unsigned char slave, unsigned int adr, double value, unsigned int
dp)
    {
        int nbw;
        int i, ivalue;
        unsigned char t[10];

                        // convertir le double en int (reel à virgule fixe)
        t[0]=slave;
        t[1]=           // a completer
         t[2]=          // a completer
        t[3]=           // a completer
        t[4]=           // a completer
         t[5]=          // a completer
                        // ecrire dans le port
        return true;
    }
```

- Compléter le programme de test (console)

```
#include <cstdlib>
#include <iostream>
#include "PortSerie.hpp"
#include "modbusRtuAscii.hpp"

using namespace std;

int main(int argc, char *argv[])
{

// création de l'objet PORT instance de la classe PortSerie 9600 bauds, 8 bits, binaire, sans
parité
// ouverture du port n°4 de l'objet PORT
// création de l'objet dynamique ReseauRTU instance de la classe ModbusRTU sur le port PORT

// écrire la valeur 57.42  dans
// fermer le port serie
// détruire le port
// détruire modbus

    system("PAUSE");
    return EXIT_SUCCESS;
}
```