Балтийский государственный технический университет  
«ВОЕНМЕХ» им. Д. Ф. Устинова

Кафедра О7 «Информационные системы и программная инженерия»

**Практическая работа №2**по дисциплине «Системное программное обеспечение»

Выполнил:  
Студент *Альков В.С.*Группа *И407Б*  
  
Преподаватель:  
*Никитин С.С.*

**Постановка задачи:**

Разработать взаимодействие клиент-сервер с помощью протокола Modbus TCP

**Задачи:**

* Формирование, отправка пакета данных со стороны клиента и получение с обратной отправкой ответа от сервера.
* Реализовать три функции: проверка соединения с сервером, вывод приветственного сообщения и функция по выбору - сложения двух чисел.

**Клиент**

Листинг программы:

#include <stdio.h>

#include <stdlib.h>

#include <WinSock2.h>

#include <malloc.h>

//используемые структуры для tcp modbus, для удобного представления и работы с пакетом

struct modbus {

unsigned short transaction\_id, protocol\_id, length;

unsigned char unit\_id;

};

struct package {

struct modbus mbus;

char data[MAXBYTE - sizeof(struct modbus)];

};

struct sum {

int x, y;

};

//функция для формаирования стандартого пакета

struct package\* getPackage(unsigned char unit\_id)

{

struct package\* pack = (struct package\*)malloc(sizeof(struct package));

pack->mbus.length = 1;

pack->mbus.unit\_id = unit\_id;

pack->mbus.transaction\_id = pack->mbus.protocol\_id = 0;

return pack;

}

//функция для формаирования пакета пинга

void ping(SOCKET sock)

{

struct package\* sendpackage = getPackage(65), recvpackage;

send(sock, (char\*)sendpackage, sizeof(modbus) + sendpackage->mbus.length - 1, NULL);

int sendresult = recv(sock, (char\*)&recvpackage, MAXBYTE, NULL);

if(sendresult != -1)

printf("Message form server: %s\n", recvpackage.data);

else

printf("timeout\n");

free(sendpackage);

}

//функция для формаирования пакета для приветствия

void sendName(SOCKET sock, char\* data, int length)

{

struct package\* sendpackage = getPackage(66), recvpackage;

strncpy((char\*)sendpackage->data, data, strlen(data)+1);

sendpackage->mbus.length = strlen(data) + 2;

send(sock, (char\*)sendpackage, sizeof(struct modbus) + sendpackage->mbus.length - 1, NULL);

int sendresult = recv(sock, (char\*)&recvpackage, MAXBYTE, NULL);

if(sendresult != -1)

printf("Message form server: %s\n", recvpackage.data);

else

printf("timeout\n");

free(sendpackage);

}

//функция создания сокета

SOCKET getSocket()

{

SOCKET sock = socket(PF\_INET, SOCK\_STREAM, IPPROTO\_TCP);

// Инициируем запрос к серверу

sockaddr\_in sockAddr;

memset (& sockAddr, 0, sizeof (sockAddr)); // Каждый байт заполняется 0

sockAddr.sin\_family = PF\_INET;

sockAddr.sin\_addr.s\_addr = inet\_addr("127.0.0.1"); //127.0.0.1

sockAddr.sin\_port = htons(502);

connect(sock, (SOCKADDR\*)&sockAddr, sizeof(SOCKADDR));

return sock;

}

//функция для формаирования пакета для нахождения суммы двух чисел

void sendSum(SOCKET sock)

{

struct package\* sendpackage = getPackage(67), recvpackage;

sendpackage->mbus.length = 1 + sizeof(struct sum);

int res;

struct sum sum;

printf("Enter 1 number: ");

scanf("%d", &sum.x);

printf("Enter 2 number: ");

scanf("%d", &sum.y);

memcpy((void\*)sendpackage->data, (void\*)&sum, sizeof(struct sum));

send(sock, (char\*)sendpackage, sizeof(struct modbus) + sendpackage->mbus.length - 1, NULL);

int sendresult = recv(sock, (char\*)&recvpackage, MAXBYTE, NULL);

if(sendresult != -1)

{

memcpy((void\*)&res, (void\*)&recvpackage.data, sizeof(int));

printf("Message form server: %d\n", res);

}

else

printf("timeout\n");

free(sendpackage);

}

int main(){

// Инициализируем DLL

WSADATA wsaData;

WSAStartup(MAKEWORD(2, 2), &wsaData);

int \_continue = 1, chose = 0;

int size = sizeof(MAXBYTE - sizeof(struct modbus));

printf("%d", size);

char name[size];

SOCKET sock;

while(\_continue)

{

sock = getSocket();

system("cls");

printf("%s\n%s\n%s\n%s\n","1. Ping", "2. Hello", "3. Sum", "4. Close");

scanf("%d",&chose);

switch(chose)

{

case 1: ping(sock); break;

case 2:

printf("Enter name: ");

scanf("%s", name);

sendName(sock, name, size);

break;

case 3: sendSum(sock); break;

case 4: \_continue = 0; break;

}

fflush(stdin);

system("pause");

}

//закрытие сокета

closesocket(sock);

// Прекращаем использование DLL

WSACleanup();

return 0;

}

**Сервер:**

Листинг программы:

#include <stdio.h>

#include <WinSock2.h>

//используемые структуры для tcp modbus, для удобного представления и работы с пакетом

struct modbus {

unsigned short transaction\_id, protocol\_id, length;

unsigned char unit\_id;

};

struct package {

struct modbus mbus;

char data[MAXBYTE - sizeof(struct modbus)];

};

struct sum {

int x, y;

};

//фунция для формирование ответа сервера на команду пинг

void onPing(struct package\* pack)

{

strcpy((char\*)pack->data, (char\*)"Server is working!");

pack->mbus.length = strlen((char\*)pack->data) + 2;

}

//фунция для формирование ответа сервера на команду приветствие

void onGreeting(struct package\* pack)

{

char str[sizeof(pack->data)];

unsigned newLen = (unsigned)(pack->mbus.length + 6) < sizeof(str) ? (unsigned)pack->mbus.length + 6 : sizeof(str) - 1;

snprintf((char\*)&str, newLen, "Hello, %s", pack->data);

snprintf((char\*)pack->data, newLen, "%s", str);

pack->mbus.length = strlen((char\*)pack->data) + 2;

}

//фунция для формирование ответа сервера на команду, сложи 2 числа

void onSum(struct package\* pack)

{

struct sum sum;

memcpy((void\*)&sum, (void\*)pack->data, pack->mbus.length-1);

int result = sum.x + sum.y;

memcpy((void\*)pack->data, (void\*)&result, sizeof(int));

pack->mbus.length = sizeof(int) + 1;

}

int main(){

// Инициализируем DLL

WSADATA wsaData;

WSAStartup( MAKEWORD(2, 2), &wsaData);

// Создаем сокет

SOCKET servSock = socket(PF\_INET, SOCK\_STREAM, IPPROTO\_TCP);

// Привязываем сокет

sockaddr\_in sockAddr;

memset (& sockAddr, 0, sizeof (sockAddr)); // Каждый байт заполняется 0

sockAddr.sin\_family = PF\_INET; // Использовать IPv4-адрес

sockAddr.sin\_addr.s\_addr = inet\_addr ("0.0.0.0"); // Определенный IP-адрес

sockAddr.sin\_port = htons (502); // Порт

bind(servSock, (SOCKADDR\*)&sockAddr, sizeof(SOCKADDR));

// Входим в состояние мониторинга

listen(servSock, 20);

SOCKADDR clntAddr;

SOCKET clntSock;

int nSize;

struct package package;

while(1){

nSize = sizeof(SOCKADDR);

clntSock = accept(servSock, (SOCKADDR\*)&clntAddr, &nSize);

// Получение клиентского запроса

recv(clntSock, (char\*)&package, MAXBYTE, NULL);

switch(package.mbus.unit\_id)

{

case 65: onPing(&package); break;

case 66: onGreeting(&package); break;

case 67: onSum(&package); break;

}

// Отправляем данные клиенту

send(clntSock, (char\*)&package, sizeof(struct modbus) + package.mbus.length - 1, NULL);

}

// Закрываем сокеты

closesocket(clntSock);

closesocket(servSock);

// Прекращаем использование DLL

WSACleanup();

return 0;

}

Результат работы программ:

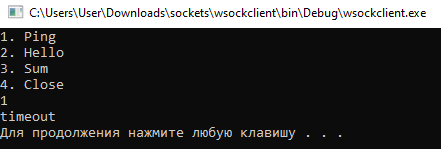


Рисунок 1 - Выполнение проверки соединения с сервером при выключенном сервере

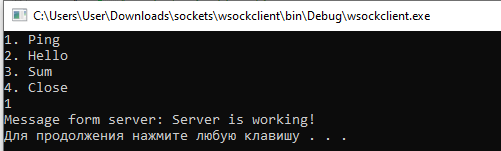


Рисунок 2 - Выполнение проверки соединения с сервером

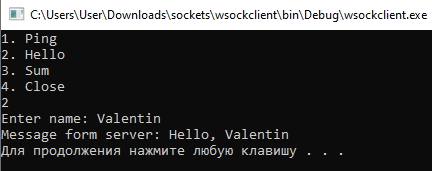


Рисунок 3 - Выполнение функции приветствия

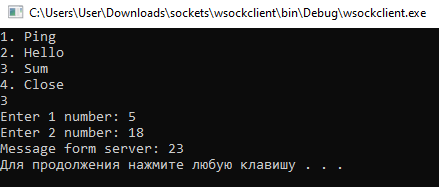


Рисунок 4 - Выполнение функции сложения

Вывод

В данной практической работе мы изучили способ взаимодействия клиент-сервер с помощью протокола Modbus TCP.