**Направление подготовки: 13.04.02 Электроэнергетика и электротехника**

**Направленность (профиль): Электродвижение и электроснабжение наземных транспортных средств**
**Уровень образования: магистратура**

**Дисциплина: "Информационные и компьютерные технологии в электротехнике"**

**Комаров В.Г. Практическое занятие 6 10.03.2026 г.**

# Телеуправление транспортным средством с использованием сокета в операционной системе Linux.

## Контрольное задание:

*Разработать и отладить программу передачи данных для управления транспортным средством с использованием сокета в операционной системе Linux согласно выданному варианту.*

**Вариант 1.**

Устройство управления транспортным средством является сервером. Программа логического управления находится на сервере. Управляющий терминал находится на удалённом компьютере и передаёт управляющие воздействия X и квитирование Z через стандартный канал по протоколу TCP/IP.

**Вариант 2.**

Устройство управления транспортным средством является сервером. Программа логического управления находится на управляющем терминале. Управляющий терминал находится на удалённом компьютере и передаёт коды режимов Y и квитирование Z через стандартный канал по протоколу TCP/IP.

**Основные понятия**

**Консоль** — простейший интерфейс ввода-вывода, относящийся к классу символьных устройств, позволяющий вводить данные с клавиатуры и выводить их на экран или другое устройство отображения. **Терминал** (от лат. terminalis – пограничный, конечный), электронное (ранние образцы – электромеханическое) устройство, используемое для организации диалогового взаимодействия пользователя с компьютером. Консоль обычно является частью компьютера, как пульт его управления, а терминал подключается через канал связи и является абонентским устройством. Если сказать по-другому, консоль - это пульт управления, а терминал - оконечное устройство связи. При этом работа консоли обычно обеспечивается прямо из ядра ОС (что не удивительно, т.к. это аппаратная часть компьютера), а для работы терминалов используется специальное программное обеспечение взаимодействия терминала с сервером по определённому протоколу. В частности для графических терминалов Linux используется система X-window.

**Се́рвер** — выделенный или специализированный компьютер для выполнения программ обработки данных и управления без оператора в автоматическом режиме. Сервер обычно работает в сетевой системе.

## Последовательность выполнения задания

Вначале разрабатываем программы в режиме *интерфейса внутренней петли* (loopback interface).

Для этого запускаем и отлаживаем программу `"TeleServer"` (файл `TS.cpp`) на одном из терминалов

```
//Программа "TeleServer" @TS.cpp

#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <cstdlib>
#include <cstddef>
#include <stdio.h>
#include <unistd.h>

int main()
{
    int sock, listener;
    struct sockaddr_in addr;
    char RxD[]="s\n"; //Принимаемые данные
      char TxD[] = "Режим установлен\n"; //Передаваемые данные
    int bytes_RxD;

    listener = socket(AF_INET, SOCK_STREAM, 0);
    if(listener < 0)
    {
        perror("socket");
        exit(1);
    }

    addr.sin_family = AF_INET;
    addr.sin_port = htons(3425);
    addr.sin_addr.s_addr = htonl(INADDR_ANY);
```

```
    if(bind(listener, (struct sockaddr *)&addr, sizeof(addr)) < 0)
    {
        perror("bind");
        exit(2);
    }

    listen(listener, 1);

    while(1)
    {
        sock = accept(listener, NULL, NULL);
        if(sock < 0)
        {
            perror("accept");
            exit(3);
        }

        while(1)
        {
            bytes_RxD = recv(sock, RxD, sizeof(RxD), 0);
            if(bytes_RxD <= 0) break;
            send(sock, TxD, sizeof(TxD), 0);
        }

      printf("ПРИНЯТЫХ БАЙТ: %d\n", sizeof(RxD));
      printf("КОД_РЕЖИМА: %s\n", RxD);
            close(sock);
    }

    return 0;
}
```

```
user1@eltron-host1:~$ g++ -o TS TS.cpp
user1@eltron-host1:~$ ./TS
```

Затем на другом терминале запускаем и отлаживаем программу `"TeleTerminal"` (файл `TT.cpp`)

```
//Программа "TeleTerminal" @TT.cpp
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <cstdlib>
#include <cstddef>
#include <stdio.h>
#include <unistd.h>

char TxD[]="s\n"; //Передаваемые данные
char RxD[32];//Принимаемые данные

int main()
{
 printf("Введите код Y исходного режима работы:\n");
 scanf("%s", TxD);
 printf("TxD = %s\n", TxD);
 printf("Количество байт = %d\n", sizeof(TxD));
    int sock;
    struct sockaddr_in addr;

    sock = socket(AF_INET, SOCK_STREAM, 0);
    if(sock < 0)
    {
        perror("socket");
        exit(1);
    }

    addr.sin_family = AF_INET;
    addr.sin_port = htons(3425); // или любой другой порт...
    addr.sin_addr.s_addr = htonl(INADDR_LOOPBACK);
    if(connect(sock, (struct sockaddr *)&addr, sizeof(addr)) < 0)
    {
        perror("connect");
        exit(2);
```

```
    }

    send(sock, TxD, sizeof(TxD), 0);
    recv(sock, RxD, sizeof(RxD), 0);

      printf("ПРИНЯТЫХ БАЙТ: %d\n", sizeof(RxD));
      printf("КВИТАНЦИЯ: %s\n", RxD);
    close(sock);

    return 0;
}
user1@eltron-host1:~$ g++ -o TT TT.cpp
user1@eltron-host1:~$ ./TT
Введите код Y исходного режима работы:
```

Вводим код исходного режима

```
s
TxD = s
Количество байт = 3
```

и получаем ответ (квитанцию) от сервера

```
ПРИНЯТЫХ БАЙТ: 32
КВИТАНЦИЯ: Режим установлен
```

На терминале сервера можем увидеть код установленного режима

```
ПРИНЯТЫХ БАЙТ: 3
КОД_РЕЖИМА: s
```

Теперь на основе программы `DriveModeLogic` и `TeleTerminal` создаём программу `TeleDrive` (файл `TD.cpp`), совмещающую логику управления режимами работы транспортного средства и передачу выходных кодов режима через канал связи серверу управления транспортным средством

```
//Программа телеуправления режимами работы транспортного средства "TeleDrive"
@TD.cpp


#include <stdio.h> // Описания стандартного ввода-вывода
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <cstdlib>
#include <cstddef>
```

```
#include <unistd.h>

int main() {
      char TxD[]="s\n"; //Передаваемые данные
      char RxD[32];//Принимаемые данные
    char y[2];
    int x=0, z=0;
    printf("Введите код Y исходного режима работы:\n");
    scanf("%s", y);
    printf("y = %s\n", y);
    printf("Введите задающий код X:\n");
    scanf("%d", &x);
    printf("x = %d\n", x);
        if(y[1]==0) {
            if(x==0) {
                  y[1]=0,z=1,printf("СТОП\n");
                              }
                  else if(x==2) {
                  y[1]=0,z=1,printf("СТОП\n");
                              }
                  else if(x==3) {
                  y[1]=0,z=1,printf("СТОП\n");
                              }
                  else if(x==4) {
                  y[1]=3,z=8,printf("ВЫБЕГ\n");
                              }
                  else if(x==1) {
                  y[1]=1,z=2,printf("ТЯГА\n");
                              }
                  else if(x==5) {
                  y[1]=1,z=2,printf("ТЯГА\n");
                              }
            else if(x==6) {
                  y[1]=2,z=4,printf("ТОРМОЗ\n");
                              }
            else if(x==7) {
```

```
                y[1]=2,z=4,printf("ТОРМОЗ\n");
                            }
          else {
                printf("ОШИБКА_ЗАДАНИЯ\n");
          return 1;         }
          }
    else {
          printf("ОШИБКА_РЕЖИМА\n");
          return 2;
          }

        printf("КОД_РЕЖИМА = %s\n", y), printf("ИНДИКАТОР_РЕЖИМА = %d\n", z);


    TxD[1]=y[1];

    printf("TxD = %s\n", TxD);

int sock;
 struct sockaddr_in addr;

 sock = socket(AF_INET, SOCK_STREAM, 0);
 if(sock < 0)
 {
     perror("socket");
     exit(1);
 }

 addr.sin_family = AF_INET;
 addr.sin_port = htons(3425); // или любой другой порт...
 addr.sin_addr.s_addr = htonl(INADDR_LOOPBACK);
 if(connect(sock, (struct sockaddr *)&addr, sizeof(addr)) < 0)
 {
     perror("connect");
     exit(2);
 }
```

```
    send(sock, TxD, sizeof(TxD), 0);
    recv(sock, RxD, sizeof(RxD), 0);

      printf("ПРИНЯТЫХ БАЙТ: %d\n", sizeof(RxD));
      printf("КВИТАНЦИЯ: %s\n", RxD);
    close(sock);

    return 0;
}
```

На одном терминале запускаем программу `TeleServer` , а на другом `TeleDrive`. Теперь с помощью программы `TeleDrive` можно дистанционно управлять транспортным средством, где установлена программа `TeleServer` , по любому каналу связи, указанному в настройках.

Например:

```
user1@eltron-host1:~$ g++ -o TD TD.cpp
user1@eltron-host1:~$ ./TD
Введите код Y исходного режима работы:
0
y = 0
Введите задающий код X:
4
x = 4
ВЫБЕГ
КОД_РЕЖИМА = 3
ИНДИКАТОР_РЕЖИМА = 8
TxD = 3
ПРИНЯТЫХ БАЙТ: 32
КВИТАНЦИЯ: Режим установлен
```

На терминале сервера можно увидеть заданный режим

```
user1@eltron-host1:~$ ./TS
ПРИНЯТЫХ БАЙТ: 3
КОД_РЕЖИМА: 3
```