БГУИР

Кафедра ЭВМ

Лабораторная работа №1

Асинхронная двунаправленная побайтная передача данных

Выполнил студент группы 850503 Проверил ассистент кафедры ЭВМ

Осетник Д.А. Глоба А.А.

Минск 2020

Цель: разработать модуль асинхронной побайтной передачи данных, соответствующий физическому уровню модели OSI, на основе последовательных интерфейсов RS-232 и RS-485.

Модуль должен быть оформлен в виде библиотеки функций или класса. Разработанный модуль будет использоваться в следующих лабораторных работах в качестве приемопередатчика.

Для физического подключения по стандарту RS-232 используются девятиконтактные разъемы.

Традиционное назначение цифровых цепей RS-232:

-SOUT – выход передатчика

-SIN – вход приемника

-RTS – сигнал-запрос от UART к модему о передачи байта

-CTS – сигнал-подтверждение от модема к UART о готовности принять байт

-DTR – сигнал от UART к модему о готовности к взаимодействию

-DSR – сигнал от модема к UART о готовности к взаимодействию

-DCD - сигнал от модема к UART об обнаружении данных

-RI – сигнал об обнаружении входящего телефонного звонка

Использование пары RTS/CTS позволяет передавать в одном направлении, а использование DSR/DTR в обратном.

Поскольку передатчик и приемник, не имеют общего источника времени, в канале используется синхросигнал. Исходя из этого выделяются два режима обмена:

1. Асинхронный – синхронизируется посылка каждого информационного байта.
2. Синхронный – синхронизируется весь информационный поток.

По своей сути, передатчик и приемник COM-порта работают как сдвиговые регистры. Cначала данные записываются параллельно в регистр, а затем при помощи сдвигов последовательно передаются при помощи синхроимпульса. Заполнение приемника так же происходит автоматически. Таким образом, если байты записываются слишком быстро, а считываются медленно, то возникает переполнение очереди передатчика. Если наоборот, то после переполнения очереди приемника происходит потеря данных.

Тактирование происходит при помощи встроенного бод-генератора. (Один бод равен одному сигналу в секунду).

Адаптер Rs-485 является продолжением стандарта RS-232. Главными различиями являются:

1. Способ передачи данных – дифференциальная пара.
2. Максимальное количество передатчика, приемников увеличено до 32.
3. Так же увеличены максимальные пропускная способность и расстояние.

void Server()

{

string name = "COM1";

HANDLE handler;

HANDLE Events[4];

char buffer[20];

string message;

cout << "Server\n\n";

Events[0] = CreateEvent(NULL, FALSE, 0, "READ");

Events[1] = CreateEvent(NULL, FALSE, 0, "WRITE");

Events[2] = CreateEvent(NULL, FALSE, 0, "EXIT");

Events[3] = CreateEvent(NULL, FALSE, 0, "WORK");

handler = CreateFile(

name.c\_str(),

GENERIC\_READ | GENERIC\_WRITE,

0,

NULL,

OPEN\_EXISTING, .

FILE\_FLAG\_OVERLAPPED,

NULL

);

if (handler == NULL)

{

printf("Error\n");

exit(GetLastError());

}

SetCommMask(handler, EV\_RXCHAR);

SetupComm(handler, 1500, 1500);

DCB ComDCM;

memset(&ComDCM, 0, sizeof(ComDCM));

ComDCM.DCBlength = sizeof(DCB);

GetCommState(handler, &ComDCM);

ComDCM.BaudRate = changeRate();

ComDCM.ByteSize = 8;

ComDCM.Parity = NOPARITY;

ComDCM.StopBits = ONESTOPBIT;

ComDCM.fAbortOnError = TRUE;

ComDCM.fDtrControl = DTR\_CONTROL\_DISABLE;

ComDCM.fRtsControl = RTS\_CONTROL\_TOGGLE;

ComDCM.fBinary = TRUE;

ComDCM.fParity = FALSE;

ComDCM.fInX = FALSE;

ComDCM.fOutX = FALSE;

ComDCM.XonChar = 0;

ComDCM.XoffChar = (unsigned char)0xFF;

ComDCM.fErrorChar = FALSE;

ComDCM.fNull = FALSE;

ComDCM.fOutxCtsFlow = FALSE;

ComDCM.fOutxDsrFlow = FALSE;

ComDCM.XonLim = 128;

ComDCM.XoffLim = 128;

if (!SetCommState(handler, &ComDCM))

{

CloseHandle(handler);

handler = INVALID\_HANDLE\_VALUE;

return;

}

}

Данный метод предназначен для инициализации COM порта при помощи структуры DCB.

В начале метода открывается порт для чтения/записи и инициализируются структуры для асинхронного чтения/записи. Далее происходит начальная настройка порта.

int read(HANDLE handler, char \*buffer)

{

int size;

DWORD numberOfBytesRead;

OVERLAPPED asynchRead;

asynchRead = { 0 };

ReadFile(handler, &size, 1 \* sizeof(int), &numberOfBytesRead, &asynchRead);

ReadFile(handler, buffer, size \* sizeof(char), &numberOfBytesRead, &asynchRead);

return size;

}

bool write(HANDLE handler, char \*buffer)

{

char symb;

int i = 0;

DWORD numberOfBytesWrite;

OVERLAPPED asynchWrite;

asynchWrite= { 0 };

while (true)

{

scanf("%c", &symb);

if (symb == '\n')

{

buffer[i] = '\0';

WriteFile(handler, &i, 1 \* sizeof(int), &numberOfBytesWrite, &asynchWrite);

WriteFile(handler, buffer, i \* sizeof(char), &numberOfBytesWrite, &asynchWrite);

if (!strcmp(buffer, "exit\0"))

return false;

else

return true;

}

else

{

buffer[i] = symb;

}

i++;

}

}

Данные методы отвечают за асинхронные чтения/запись данных.

Чтение/запись происходит при помощи функций ReadFile/WriteFile.

Вывод: в ходе данной лабораторной работы был разработан программный модуль (в виде класса на языке C++) для асинхронной побайтной передачи данных.