Вот пример кода, который получает звук из динамика используя порты:

MODEL SMALL

STACK 256

CODESEG

Start:

mov ax,@data ; установка в ds адpеса

mov ds,ax ; сегмента данных

in al,61H ; получить текущее значение

or al,02H ; установить второй бит

out 61H, al ; вывести значение в порт

mov cx,9000H ; количество циклов

delay:

loop delay ; ждать

and al,0fdH ; очистить второй бит

out 61H, al ; вывести в порт

Exit:

CALL ExitProgramm

INCLUDE function.asm

End Start

После запуска этой программы Вы должны услышать короткий звук из динамика.

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

Вставка кода на ассемблере в программу на с:

#include <stdio.h>

int main(void)

{

int a = 1, b = 2, c = 0;

\_\_asm {

mov eax, a;

mov ebx, b;

add eax, ebx;

mov c, eax;

}

printf("%x + %x = %x\n", a, b, c);

return 0;

}

+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

Последовательные порты полюбились разработчикам за их простоту в обслуживании и использовании.

И конечно же писать в консоль терминальной программы это всё хорошо, но хочеться своё приложение, которое по нажатии клавиши на экране выполняет нужные вам действия ;)

В этой статье опишу *как работать с com портом на языке Си++*.

Решение простое, но почемуто рабочий пример найден был не сразу. За сим сохраняю его тут.

Конечно вы можете использовать кроссплатформенные решения вроде QSerial - библиотеки в составе Qt, я наверное так и сделаю, но в будующем. Сейчас же речь о "чистом" виндовском *C++*. Писать будем в Visual Studio. У меня 2010, хотя роли это никакой не играет...

Создаём новый консольный Win32 проект.

Инклудим header файлы:

[?](http://blablacode.ru/programmirovanie/392)

|  |  |
| --- | --- |
| 1  2  3 | #include <windows.h>  #include <iostream>  using namespace std; |

Объявляем обработчик com порта:

[?](http://blablacode.ru/programmirovanie/392)

|  |  |
| --- | --- |
| 1 | HANDLE hSerial; |

Я делаю это глобально, чтобы не заморачиваться с указателями при передаче его в функции.

Дальше начинаем формировать функцию main:

[?](http://blablacode.ru/programmirovanie/392)

|  |  |
| --- | --- |
| 1  2 | int \_tmain(int argc, \_TCHAR\* argv[])  { |

Терпеть не могу виндовский стиль программирования. Обозвали всё посвоему и сидят радуются...

Теперь магия объявления строки с именем порта. Дело в том, что char оно преобразовывать само не умеет.

[?](http://blablacode.ru/programmirovanie/392)

|  |  |
| --- | --- |
| 1 | LPCTSTR sPortName = L"COM1"; |

Работа с последоавательными портами в Windows проходит как с файлом. Открываем первый *ком порт для записи/чтения*:

[?](http://blablacode.ru/programmirovanie/392)

|  |  |
| --- | --- |
| 1 | hSerial = ::CreateFile(sPortName,GENERIC\_READ | GENERIC\_WRITE,0,0,  OPEN\_EXISTING,FILE\_ATTRIBUTE\_NORMAL,0); |

Проверяем работоспособность:

[?](http://blablacode.ru/programmirovanie/392)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | if(hSerial==INVALID\_HANDLE\_VALUE)  {          if(GetLastError()==ERROR\_FILE\_NOT\_FOUND)      {          cout << "serial port does not exist.\n";      }      cout << "some other error occurred.\n";  } |

Теперь нужно настроить параметры соединения:

[?](http://blablacode.ru/programmirovanie/392)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | DCB dcbSerialParams = {0};  dcbSerialParams.DCBlength=sizeof(dcbSerialParams);  if (!GetCommState(hSerial, &dcbSerialParams))  {      cout << "getting state error\n";  }  dcbSerialParams.BaudRate=CBR\_9600;  dcbSerialParams.ByteSize=8;  dcbSerialParams.StopBits=ONESTOPBIT;  dcbSerialParams.Parity=NOPARITY;  if(!SetCommState(hSerial, &dcbSerialParams))  {      cout << "error setting serial port state\n";  } |

На msdn советуют сначала получить параметры, а затем менять их. Мы ещё только учимся, поэтому делаем как просят.

Теперь объявим строку, которую будем передавать и переменные необходимые для этого:

[?](http://blablacode.ru/programmirovanie/392)

|  |  |
| --- | --- |
| 1  2  3 | char data[] = "Hello from C++";  // строка для передачи  DWORD dwSize = sizeof(data);   // размер этой строки  DWORD dwBytesWritten;    // тут будет количество собственно переданных байт |

Посылаем строку. Напомню, что пример простейший, поэтому никаких особо проверок я не делаю:

[?](http://blablacode.ru/programmirovanie/392)

|  |  |
| --- | --- |
| 1 | BOOL iRet = WriteFile (hSerial,data,dwSize,&dwBytesWritten,NULL); |

Также я решил вывести для контроля размер строки и количество отосланных байт:

[?](http://blablacode.ru/programmirovanie/392)

|  |  |
| --- | --- |
| 1 | cout << dwSize << " Bytes in string. " << dwBytesWritten << " Bytes sended. " << endl; |

В конце программы делаем бесконечный цикл чтения данных:

[?](http://blablacode.ru/programmirovanie/392)

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | while(1)      {          ReadCOM();      }      return 0;  } |

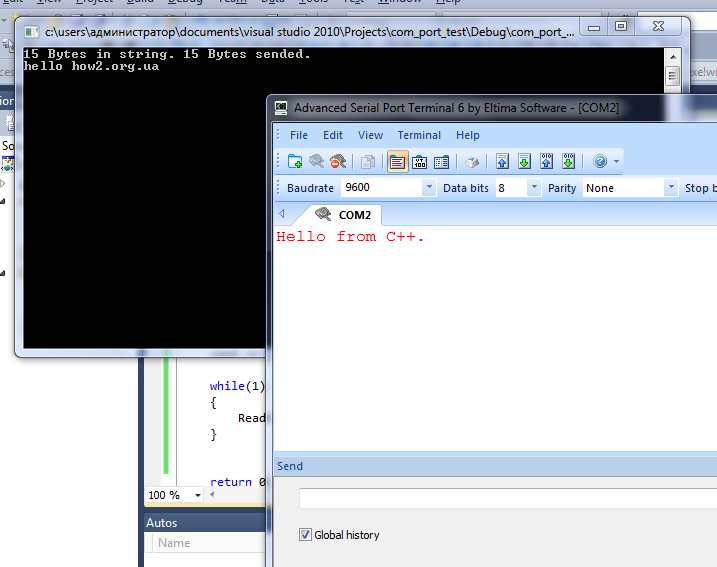
Теперь функция чтения:

[?](http://blablacode.ru/programmirovanie/392)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | void ReadCOM()  {        DWORD iSize;        char sReceivedChar;        while (true)        {              ReadFile(hSerial, &sReceivedChar, 1, &iSize, 0);  // получаем 1 байт              if (iSize > 0)   // если что-то принято, выводим                  cout << sReceivedChar;        }  } |

Вот собственно и весь пример.

создан [виртуальный com порт](http://blablacode.ru/%D0%BC%D0%B8%D0%BA%D1%80%D0%BE%D0%BA%D0%BE%D0%BD%D1%82%D1%80%D0%BE%D0%BB%D0%BB%D0%B5%D1%80%D1%8B/%D0%B2%D0%B8%D1%80%D1%82%D1%83%D0%B0%D0%BB%D1%8C%D0%BD%D1%8B%D0%B9-com-%D0%BF%D0%BE%D1%80%D1%82-%D1%88%D0%BB%D1%91%D0%BC-%D0%B8%D0%B7-com1-%D0%B2-com2-%D0%B8-%D0%B2-%D0%BF%D1%80%D0%BE%D0%B5%D0%BA%D1%82-isis.html). И высылаем из COM1 в COM2:



Из нашей программы было отправлено "Hello from C++", а из терминала "hello how2.org.

+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

**HANDLE hCOM=CreateFile("COM1",GENERIC\_WRITE,0,NULL,OPEN\_EXISTING,FILE\_ATTRIBUTE\_NORMAL,NULL);**

**if (hCOM!=INVALID\_HANDLE\_VALUE)**

**{**

**cout << "COM1 is open OK!" << endl;**

**char buffer[30];**

**memset(buffer,0,sizeof(buffer));**

**strcpy(buffer,"SAVE TO COM1");**

**DWORD nb;**

**OVERLAPPED ov;**

**WriteFile(hCOM,buffer,sizeof(buffer),&nb,&ov);**

**CloseHandle(hCOM);**

**}**

**else cout << "Error Open COM1" << endl;**

**...**

**...**

Ну вот и всё, приложение готово.

+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

Открытие порта

С последовательными и параллельными портами в [Win32](https://ru.wikibooks.org/w/index.php?title=Win32&action=edit&redlink=1) работают как с [файлами](https://ru.wikibooks.org/w/index.php?title=%D0%A4%D0%B0%D0%B9%D0%BB&action=edit&redlink=1). Для открытия порта используется функция CreateFile. Эта функция предоставляется [Win32](https://ru.wikibooks.org/w/index.php?title=Win32&action=edit&redlink=1)[API](https://ru.wikibooks.org/w/index.php?title=API&action=edit&redlink=1). Ее прототип выглядит так:

HANDLE CreateFile(

LPCTSTR lpFileName,

DWORD dwDesiredAccess,

DWORD dwShareMode,

LPSECURITY\_ATTRIBUTES lpSecurityAttributes,

DWORD dwCreationDistribution,

DWORD dwFlagsAndAttributes,

HANDLE hTemplateFile

);

**lpFileName**

[Указатель](https://ru.wikibooks.org/w/index.php?title=%D0%A3%D0%BA%D0%B0%D0%B7%D0%B0%D1%82%D0%B5%D0%BB%D1%8C&action=edit&redlink=1) на [строку](https://ru.wikibooks.org/w/index.php?title=%D0%A1%D1%82%D1%80%D0%BE%D0%BA%D0%B0&action=edit&redlink=1) с именем открываемого или создаваемого [файла](https://ru.wikibooks.org/w/index.php?title=%D0%A4%D0%B0%D0%B9%D0%BB&action=edit&redlink=1). [Формат](https://ru.wikibooks.org/w/index.php?title=%D0%A4%D0%BE%D1%80%D0%BC%D0%B0%D1%82_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)&action=edit&redlink=1) этой строки может быть очень «хитрым». В частности можно указывать сетевые имена для доступа к файлам на других компьютерах. Можно открывать [логические разделы](https://ru.wikibooks.org/w/index.php?title=%D0%9B%D0%BE%D0%B3%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D1%80%D0%B0%D0%B7%D0%B4%D0%B5%D0%BB&action=edit&redlink=1) или [физические диски](https://ru.wikibooks.org/w/index.php?title=%D0%A4%D0%B8%D0%B7%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%B4%D0%B8%D1%81%D0%BA&action=edit&redlink=1) и работать в обход [файловой системы](https://ru.wikibooks.org/wiki/%D0%A4%D0%B0%D0%B9%D0%BB%D0%BE%D0%B2%D0%B0%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0).

Последовательные порты имеют имена "COM1", "COM2", "COM3", "COM4", "COM5", "COM6", "COM7", "COM8", "COM9". Для доступа к портам, чей номер больше 9, необходимо указывать имя порта как "\\.\COMx", где x - номер порта. Например, "\\.\COM72" (в нотации языка C/C++ строка будет выглядеть "\\\\.\\COM72"). Такой синтаксис подходит для любого номера порта. Точно так же они назывались в [MS-DOS](https://ru.wikibooks.org/w/index.php?title=MS-DOS_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)&action=edit&redlink=1). [Параллельные порты](https://ru.wikibooks.org/w/index.php?title=%D0%9F%D0%B0%D1%80%D0%B0%D0%BB%D0%BB%D0%B5%D0%BB%D1%8C%D0%BD%D1%8B%D0%B9_%D0%BF%D0%BE%D1%80%D1%82&action=edit&redlink=1) называются «LPT1», «LPT2» и так далее.

**dwDesiredAccess**

Задает тип [доступа](https://ru.wikibooks.org/w/index.php?title=%D0%94%D0%BE%D1%81%D1%82%D1%83%D0%BF&action=edit&redlink=1) к [файлу](https://ru.wikibooks.org/w/index.php?title=%D0%A4%D0%B0%D0%B9%D0%BB&action=edit&redlink=1). Возможно использование следующих значений:

* 0 Опрос атрибутов устройства без получения доступа к нему.
* GENERIC\_READ Файл будет считываться.
* GENERIC\_WRITE Файл будет записываться.
* GENERIC\_READ|GENERIC\_WRITE Файл будет и считываться и записываться.

**dwShareMode**

Задает параметры [совместного доступа](https://ru.wikibooks.org/w/index.php?title=%D0%A1%D0%BE%D0%B2%D0%BC%D0%B5%D1%81%D1%82%D0%BD%D1%8B%D0%B9_%D0%B4%D0%BE%D1%81%D1%82%D1%83%D0%BF&action=edit&redlink=1) к [файлу](https://ru.wikibooks.org/w/index.php?title=%D0%A4%D0%B0%D0%B9%D0%BB&action=edit&redlink=1). Коммуникационные порты нельзя делать разделяемыми, поэтому данный параметр должен быть равен 0.

**lpSecurityAttributes**

Задает [атрибуты](https://ru.wikibooks.org/w/index.php?title=%D0%90%D1%82%D1%80%D0%B8%D0%B1%D1%83%D1%82&action=edit&redlink=1) защиты файла. Поддерживается только в [Windows NT](https://ru.wikibooks.org/w/index.php?title=Windows_NT_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)&action=edit&redlink=1" \o "Windows NT (программирование) (страница не существует)). Однако при работе с портами должен в любом случае равняться NULL.

**dwCreationDistribution**

Управляет [режимами](https://ru.wikibooks.org/w/index.php?title=%D0%A0%D0%B5%D0%B6%D0%B8%D0%BC%D0%B0%D0%BC&action=edit&redlink=1) автосоздания, автоусечения файла и им подобными. Для коммуникационных портов всегда должно задаваться OPEN\_EXISTING.

**dwFlagsAndAttributes**

Задает атрибуты создаваемого файла. Также управляет различными режимами [обработки](https://ru.wikibooks.org/w/index.php?title=%D0%9E%D0%B1%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA&action=edit&redlink=1). При работе с портом этот параметр должен быть или равным 0, илиFILE\_FLAG\_OVERLAPPED. Нулевое значение используется при [синхронной работе](https://ru.wikibooks.org/w/index.php?title=%D0%A1%D0%B8%D0%BD%D1%85%D1%80%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%B0&action=edit&redlink=1) с портом, а FILE\_FLAG\_OVERLAPPED при [асинхронной](https://ru.wikibooks.org/w/index.php?title=%D0%90%D1%81%D0%B8%D0%BD%D1%85%D1%80%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%B0&action=edit&redlink=1), или, другими словами, при фоновой обработке [ввода](https://ru.wikibooks.org/w/index.php?title=%D0%92%D0%B2%D0%BE%D0%B4&action=edit&redlink=1)/[вывода](https://ru.wikibooks.org/w/index.php?title=%D0%92%D1%8B%D0%B2%D0%BE%D0%B4&action=edit&redlink=1). Подробнее про асинхронный ввод/вывод я расскажу позже.

**hTemplateFile**

Задает описатель файла-[шаблона](https://ru.wikibooks.org/w/index.php?title=%D0%A8%D0%B0%D0%B1%D0%BB%D0%BE%D0%BD&action=edit&redlink=1" \o "Шаблон (страница не существует)). При работе с портами всегда должен быть равен NULL.

При успешном открытии файла, в данном случае порта, [функция](https://ru.wikibooks.org/w/index.php?title=%D0%A4%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D1%8F_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)&action=edit&redlink=1) возвращает [дескриптор](https://ru.wikibooks.org/w/index.php?title=%D0%94%D0%B5%D1%81%D0%BA%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D1%80_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)&action=edit&redlink=1) ([HANDLE](https://ru.wikibooks.org/w/index.php?title=HANDLE_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)&action=edit&redlink=1)) файла. При ошибке [[|INVALID HANDLE VALUE]]. Код [ошибки](https://ru.wikibooks.org/w/index.php?title=%D0%9E%D1%88%D0%B8%D0%B1%D0%BA%D0%B0&action=edit&redlink=1) можно получитить вызвав функцию [[|GetLastError]].

Закрытие порта

Открытый порт должен быть закрыт перед завершением работы [программы](https://ru.wikibooks.org/w/index.php?title=%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B0&action=edit&redlink=1). В [Win32](https://ru.wikibooks.org/w/index.php?title=Win32_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)&action=edit&redlink=1) закрытие объекта по его [дескриптору](https://ru.wikibooks.org/w/index.php?title=%D0%94%D0%B5%D1%81%D0%BA%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D1%80&action=edit&redlink=1) выполняет функция CloseHandle:

BOOL CloseHandle(

HANDLE hObject

);

При успешном завершении функция возвращает не нулевое значение, при ошибке нуль.

**Пример открытия/закрытия на языке**[**C**](https://ru.wikibooks.org/w/index.php?title=%D0%A1%D0%B8_(%D1%8F%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F)&action=edit&redlink=1)

#include *<windows.h>*

*//. . .*

HANDLE Port;

*//. . .*

Port = CreateFile("**\\\\**.**\\**COM2", GENERIC\_READ | GENERIC\_WRITE, 0, NULL, OPEN\_EXISTING, 0, NULL);

**if** (Port == INVALID\_HANDLE\_VALUE) {

MessageBox(NULL, "Невозможно открыть последовательный порт", "Error", MB\_OK);

ExitProcess(1);

}

*//. . .*

CloseHandle(Port);

*//. . .*

В данном [примере](https://ru.wikibooks.org/w/index.php?title=%D0%9F%D1%80%D0%B8%D0%BC%D0%B5%D1%80&action=edit&redlink=1) открывается порт СОМ2 для [чтения](https://ru.wikibooks.org/w/index.php?title=%D0%A7%D1%82%D0%B5%D0%BD%D0%B8%D0%B5&action=edit&redlink=1) и [записи](https://ru.wikibooks.org/w/index.php?title=%D0%97%D0%B0%D0%BF%D0%B8%D1%81%D1%8C&action=edit&redlink=1), используется синхронный режим обмена. Проверяется успешность открытия порта, при ошибке выводится сообщение и программа завершается. Если порт открыт успешно, то он закрывается.