Министерство образования и науки, молодежи и спорта Украины
Севастопольский национальный технический университет

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

Лабораторная работа №6
«Разработка приложений для работы с *COM*-портом в среде C++*Builder*»
по дисциплине
«**Персональные компьютеры в вычислительной практике**»
для студентов дневной и заочной формы обучения
по направлению 6.050901 — Радиотехника.

Севастополь
2013 г.

УДК.519.72

Методические указания «Лабораторная работа №6 Разработка приложений для работы с *COM*-портом в среде C++*Builder»* по дисциплине «Персональные компьютеры в вычислительной практике» для студентов дневной и заочной форм обучения по направлению 6.050901 — Радиотехника / Сост. А.В. Лукьянчиков, И.В. Сердюк — Севастополь: Изд-во СевНТУ, 2013. — 15 с.

Методические указания рассмотрены и утверждены на заседании кафедры радиотехники и телекоммуникаций (протокол № _ от __ ____ 2013 г.)

Допущено учебно-методическим центром СевНТУ в качестве методических указаний.

Рецензент: канд. техн. наук, доцент Савочкин А.А.

Ответственный за выпуск: заведующий кафедрой радиотехники и телекоммуникаций, доктор техн. наук, профессор Гимпилевич Ю.Б.,

6. ЛАБОРАТОРНАЯ РАБОТА № 6
РАЗРАБОТКА ПРИЛОЖЕНИЙ ДЛЯ РАБОТЫ С *COM*-ПОРТОМ В СРЕДЕ C++*BUILDER*

6.1. Цель работы: На примере разработанного приложения в среде *C++ Builder* разобраться в принципах обмена информацией между компьютером и внешним устройством посредством *COM*-порта.

6.2. Теоретические сведения

Последовательный интерфейс для передачи данных в одну сторону использует одну сигнальную линию, по которой информационные биты передаются друг за другом последовательно. Такой способ передачи определяет название интерфейса и порта, его реализующего (*Serial Interface* и *Serial Port*). Последовательная передача данных может осуществляться в синхронном и асинхронном режимах.

При асинхронной передачи каждому байту предшествует старт-бит, сигнализирующий приемнику о начале очередной посылки, за которой следуют биты данных или бит паритета (конроля четности). Завершает посылку стоп-бит. Старт-бит (имеющий значение лог. "0") следующего посланного байта может посылаться в любой момент после окончания стоп-бита. Старт-бит обеспечивает механизм синхронизации приемника по сигналу от передатчика. Внутренний генератор синхронизации приемника использует счетчик-делитель опорной частоты, обнуляемый в момент приема начала старт-бита. Этот счетчик генерирует внутренние стробы, по которым приемник фиксирует последующие принимаемые биты.

Формат асинхронной посылки позволяют выявить возможные ошибки передачи.

Для асинхронного режима принят ряд стандартных скоростей обмена: 50, 75, 110, 150, 300, 600, 1200, 2400, 4800, 19200, 38400, 57600, 115200 бит/сек. Количество бит данных может составлять 5, 6, 7, 8 бит. Количество стоп битов может быть 1, 1.5, 2 бита. Асинхронный режим работы в PC реализуется с помощью COM-порта с использованием протокола *RS*-232*C*.

Синхронный режим передачи предполагает постоянную активность канала связи. Посылка начинается с синхробайта, за которым плотно следует поток информационных бит. Если у передатчика нет данных для передачи, он заполняет паузу непрерывной посылкой байтов синхронизации. При передаче больших массивов данных накладные расходы на синхронизацию в данном режиме необходима будет ниже, чем в асинхронном. Однако в синхронном режиме необходима внешняя синхронизация приемника с передатчиком, поскольку даже малое отклонение частот приведет к быстро накапливающейся ошибке и искажению принимаемых данных. Внешняя синхронизация возможна либо с помощью отдельной линии передачи для передачи сигнала синхронизации, либо с использованием самосинхронизирующего кодирования данных, при котором на приемной стороне из принятого сигнала могут быть и импульсы синхронизации. В любом случае синхронный режим требует либо дорогих линий связи, либо дорогого оконченного оборудования. Для PC существуют специальные платы — адаптеры *SDLC*, поддерживающие синхронный режим обмена. Они используются в основном для связи с большими машинами *IBM* и в настоящее время мало распространены. Из синхронных адаптеров в настоящее время чаще всего применяются адаптеры интерфейса *V*.35.

Последовательный интерфейс на физическом уровне может иметь различные реализации, различающиеся способом передачи электрических сигналов. Существует ряд родственных международных стандартов: *RS*-232*C*, *RS*-432*A*, *RS*-422*A*, *RS*485.

Несимметричные линии интерфейсов *RS*-232*C*, *RS*-432*A* имеют самую низкую защищенность от синфазной помехи. Лучшие параметры имеет двухточечный интерфейс

RS-422*A* и его магистральный (шинный) родственник *RS*-485, работающие на симметричных линиях связи. В них для каждого сигнала используются дифференциальные сигналы с отдельной (витий) парой приводов.

Наибольшее распространение в PC получил простейший из этих — стандарт *RS*-232*C*. В промышленной автоматике широко применяется *RS*-422A, а также *RS*-485, встречающийся и в

некоторых принтерах. Существуют относительно несложные преобразователи сигналов для согласования всех этих интерфейсов.

6.2.1. Интерфейс *RS*-232*C*

Интерфейс *RS*-232*C* предназначен для подключения аппаратуры, передающей или принимающей данные (ООД — оконечное оборудование данных, или АПД — аппаратура передачи данных; *DTE — Data Terminal Equipment*), к оконечной аппаратуре каналов данных (АКД; *DCE — Data Communication Equipment*). В роли АПД может выступать компьютер, принтер, плоттер и другое периферийное оборудование. В роли АКД обычно выступает модем. Конечной целью подключения является соединение двух устройств АПД. Полная схема соединения приведена на рис. 1; интерфейс позволяет исключить канал удаленной связи вместе с парой устройств АКД, соединив устройства непосредственно с помощью нуль-модемного кабеля (рис. 2).

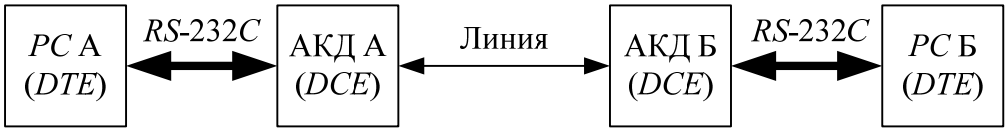

Рисунок 1 — Полная схема соединения по *RS*-232*C*

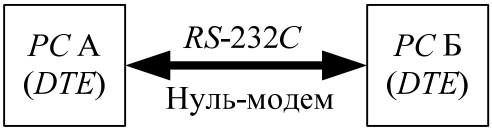

Рисунок 2 — Соединение по *RS*-232*C* нуль-модемным кабелем

Стандарт описывает управляющие сигналы интерфейса, пересылку данных, электрический интерфейс и типы разъемов. Стандарт описывает синхронный и асинхронный режимы обмена, но COM-порты поддерживают только асинхронный режим. Функционально *RS*-232*C* эквивалентен стандарту МККТТ *V*.24/*V*.28 и стыку С2, но они имеют различные названия одних и тех же используемых сигналов.

6.2.2. Электрический интерфейс

Стандарт *RS*-232*C* изображенный на рисунке 3, использует несимметричные передатчики и приемники — сигнал передается относительно общего провода — схемной земли *с*имметричные дифференциальные сигналы используются в других интерфейсах — например,*RS*-422).

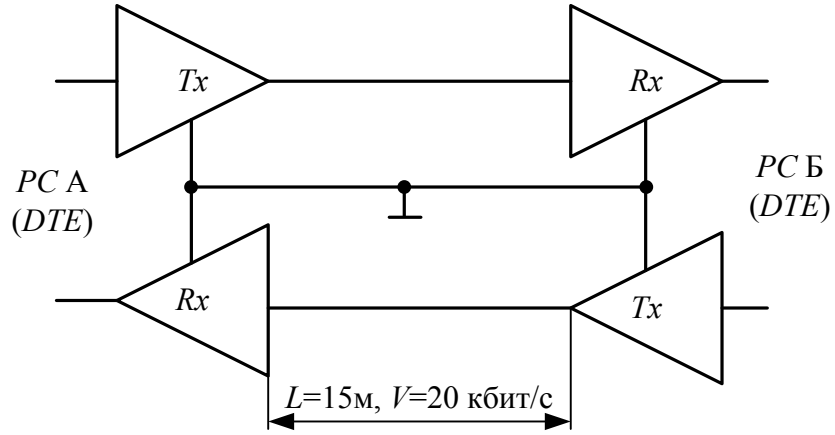

Рисунок 3 — Стандарт последовательного интерфейса.

Интерфейс НЕ ОБЕСПЕЧИВАЕТ ГАЛЬВАНИЧЕСКОЙ РАЗВЯЗКИ устройств. Логической единице соответствует уровень напряжения на входе приемника в диапазоне (–12...–3) В. Для линий

управляющих сигналов это состояние называется *ON* ("включено"), для линий последовательных данных называется *MARK*. Логическому нулю соответствует напряжение в диапазоне (+3...+12) В. Для линий управляющих сигналов это состояние называется *OFF* ("выключено"), для линий последовательных данных называется *SPACE*. Между уровнями (–3...+3)В имеется зона нечувствительности, обусловливающая гистерезис приемника: состояние линии будет считаться измененным только после пересечения соответствующего порога.

Уровни сигналов на выходах передатчиков должны быть в диапазонах (–12...–5) В и (+5...+12) В для представления единицы и нуля соответственно. Разность потенциалов между схемными землями (*SG*) соединяемых устройств должна быть менее 2 В, при более высокой разности потенциалов возможно неверное восприятие сигналов.

Интерфейс предполагает наличие ЗАЩИТНОГО ЗАЗЕМЛЕНИЯ для соединяемых устройств, если они оба питаются от сети переменного тока и имеют сетевые фильтры.

Подключение и отключение интерфейсных кабелей устройств с автономным питанием (не питающихся от интерфейса, таких как, например, мышь) должно производиться при отключении питания. В противном случае разность не выровненных потенциалов устройств в момент коммутации (присоединения или отсоединения разъема) может оказаться приложенной к выходным или входным (что опаснее) цепям интерфейса и вывести из строя микросхемы.

Для интерфейса *RS*-232*C* специально выпускаются буферные микросхемы приемников (с гистерезисом) и передатчиков двуполярного сигнала. При несоблюдении правил заземления и коммутации включенных устройств они обычно являются первыми (хорошо, если единственными) жертвами "пиротехнических"эффектов. Иногда их устанавливают в "кроватках", что сильно облегчает замену. Часто буферные схемы входят прямо в состав интерфейсных БИС. Это удешевляет изделие, экономит место на плате, но в случае аварии обычно оборачивается крупными финансовыми потерями. Вывести из строя интерфейсные микросхемы замыканием сигнальных цепей маловероятно, поскольку ток короткого замыкания передатчиков обычно ограничен на уровне 20 мА.

Стандарт *RS*-232*C* регламентирует типы применяемых разъемов, что обеспечивает высокий уровень совместимости аппаратуры различных производителей.

На аппаратуре *DTE* (в том числе, и на COM-портах *PC*) принято устанавливать вилки, (*male* — "папа") *DB*25-*P* или более компактный вариант — *DB*9-*P*.

Девятиштырьковые разъемы не имеют контактов для дополнительных сигналов, необходимых для синхронного режима (в большинстве 25тырьковых разъемов эти контакты не используются).

На аппаратуре *DCE* (модемах) устанавливают розетки (*female* — "мама") *DB*25-*S* или *DB*-9*S*.

Это правило предполагает, что разъемы *DCE* могут подключаться к разъемам *DTE* непосредственно (если позволяет геометрия конструктива) или через переходные "прямые" кабели с розеткой и вилкой, у которых контакты *с*оединены "один в один". Переходные кабели могут являться и переходниками с 9 на 25-штырьковые разъемы.

Если аппаратура *DTE* соединяется без модемов, то разъемы устройств (вилки) соединяются между собой нуль-модемным кабелем (*Zero-modem* или *Z-modem*), имеющим на обоих концах розетки, контакты которых соединяются перекрестно.

Если на каком-либо устройстве *DTE* (принтер, плоттер, дигитайзер) установлена розетка - это почти стопроцентный признак того, что к другому устройству (компьютеру) оно должно подключаться прямым кабелем, аналогичным кабелю подключения модема. Розетка устанавливается обычно на тех устройствах, у которых удаленное подключение через модем не предусмотрено (или бессмысленно, как, например, у дигитайзера).

В табл.1 приведено назначение контактов разъемов COM-портов (и любой другой аппаратуры *DTE*). Назначение контактов разъема *DB*25*S* определено стандартом *EIA*/*TIA*-232-E, разъем *DB*9*S* определен стандартом *EIA*/ *TIA*-574.

У модемов (*DCE*) название цепей и назначение контактов, естественно, совпадает, но роли сигналов (вход-выход) меняются на противоположные.

Таблица 1 — Разъемы и сигналы интерфейса *RS*-232*C*

Обозначение цепи		Контакт разъема		Провод шлейфа выносного разъема *PC*				Направление	Название цепи
RS-232	Стык 2*D*	*DB25S*	*DB9S*	1*	2*	3*	4*	*I*/*O*	
PG	101	1	—	(10)	(10)	(10)	1	—	*Protected Ground* — Защитная земля
TD	103	2	3	3	5	3	3	*O*	*Transmit Data* — Передаваемые данные
RD	104	3	2	2	3	4	5	*I*	*Receive Data* — Принимаемые данные
RTS	105	4	7	7	4	8	7	*O*	*Request to Send* — Запрос передачи
CTS	106	5	8	8	6	7	9	*I*	*Clear to Send* — Готовность модема к приему данных для передачи
DSR	107	6	6	6	2	9	11	*I*	*Data set ready* — Готовность модема к работе
SG	102	7	5	5	9	1	13	—	*Signal ground* — Сигнальная земля
DCD	109	8	1	1	1	5	15	*I*	*Data carrier detect* — Несущая обнаружена
DCR	108/2	20	4	4	7	2	14	*O*	*Data terminal ready* — Готовность терминала к работе
RI	125	22	9	9	8	6	18	*I*	*Ring indicator* — Индикатор вызова

1* — шлейф 8-битных мультикарт.

2* — шлейф 16-битных мультикарт и портов на системных платах.

3* — вариант шлейфа портов на системных платах.

4* — широкий шлейф к 25-контактному разъему.

Подмножество сигналов *RS*-232*C*, относящихся к асинхронному режиму, рассмотрим с точки зрения COM-порта *PC*, являющегося по терминологии *RS*-232*C* терминалом данных (*DTE*). Следует помнить, что активному состоянию сигнала ("включено") и логической единице передаваемых данных соответствует отрицательный потенциал (ниже –3 В) сигнала интерфейса, а состоянию "выключено" и логическому нулю - положительный (выше +3 В). Назначение сигналов интерфейса приведено в табл.2.

Таблица 2 — Назначение сигналов интерфейса

Сигнал	Назначение
PG	*Protected Ground* — защитная земля, соединяется с корпусом устройства и экраном кабеля
SG	*Signal Ground* — сигнальная (схемная) земля, относительно которой действуют уровни сигналов
TD	*Transmit Data* —последовательные данные — выход передатчика
RD	*Receive Data* — последовательные данные — вход приемника
RTS	*Request To Send* — выход запроса передачи данных: состояние “включено” уведомляет модем о наличии у терминала данных для передачи. В полудуплексном режиме используется для управления направлением — состояние “включено” служит сигналом модему на переключение в режим передачи
CTS	*Clear To Send* — вход разрешения терминалу передавать данные. Состояние “выключено” запрещает передачу данных. Сигнал используется для аппаратного управления потоками данных
DSR	*Data Set Ready* — вход сигнала готовности от аппаратуры передачи данных (модем в рабочем режиме подключен к каналу и закончил действия по согласованию с аппаратурой на противоположном конце канала)
DTR	*Data Terminal Ready* — выход сигнала готовности терминала к обмену данными. Состояние “включено” поддерживает коммутируемый канал в состоянии соединения
DCD	*Data Carrier Detected* — вход сигнала обнаружения несущей удаленного модема
RI	*Ring Indicator* — вход индикатора вызова (звонка). В коммутируемом канале этим сигналом модем сигнализирует о принятии вызова

6.2.3. Ресурсы COM-портов

Начиная с первых моделей в *PC* имелся последовательный интерфейс — COM-порт (*Communications Port* — коммуникационный порт). Этот порт обеспечивает асинхронный обмен по стандарту *RS*-232*C*. Компьютер может иметь до четырех последовательных портов COM1…*COM*4 (для машин класса *AT* типично наличие двух портов). COM-порты имеют внешние разъемы-вилки (*Male* "папа") *DB*25*P* или *DB*9*P*, выведенные на заднюю панель компьютера (назначение выводов приведено в табл.1)

COM-порты реализуются на микросхемах *UART*, совместимых с семейством 18250. Они занимают в пространстве ввода/вывода по 8 смежных 8-битных регистров и могут располагаться по стандартным базовым адресам 3*F*8*h* (*COM*1), 2*F*8*h* (*COM*2), 3*E*8*h* (*COM*3), 2*E*8*h* (*COM*4). Для портов *COM*3 и *COM*4 возможны альтернативные адреса 3*EOh*, 338*h* и 2*EOh*, 238*h* соответственно. Для *PS*/2 стандартными для портов COM3-COM8 являются адреса 3220*h*, 3228*h*,4220*h*, 4228*h*, 5220*h* и 5228*h* соответственно.

Порты могут вырабатывать аппаратные прерывания *IRQ*4 (обычно используются для *COM*1 и *COM*3) и *IRQ*3 (для *COM*2 и *COM*4). Кроме того, возможно использование линий прерываний *IRQ*11 (вместо *IRQ*4) и *IRQ*10 (вместо *IRQ*3). Возможность разделяемого использования одной линии запроса несколькими портами (или ее разделения с другими устройствами) зависит от реализации аппаратного подключения и программного обеспечения. При использовании портов, установленных на шину *ISA*, разделяемые прерывания обычно не работают.

6.2.4. Конфигурирование COM-портов

Управление последовательным портом разделяется на два этапа - предварительное конфигурирование (*Setup*) аппаратных средств порта и текущее (оперативное) переключение режимов работы прикладным или системным ПО. Способ и возможности конфигурирования COM-портов зависят от его исполнения и местоположения. Порт, расположенный на плате расширения (обычно на мультикарте), устанавливаемой в слот *ISA* или *ISA+VLB*, обычно конфигурируется джамперами на самой плате. Порт, расположенный на системной плате, обычно конфигурируется через *BIOS Setup*.

Конфигурированию подлежат следующие параметры:

* Базовый адрес, который может иметь значение 3*F*8*h*, 2*F*8*h*, 3*E*8*h* (3*EOh*,338*h*), 2*E*8*h* (2*EOh*, 238*h*). При инициализации *BIOS* проверяет наличие портов по адресам именно в этом порядке и, соответственно, присваивает обнаруженным портам логические имена COM1, *COM*2, *COM*3 и *COM*4.

* Используемая линия запроса прерывания: для COM1 и *COM*3 обычно используется *IRQ*4 или *IRQ*11, для *COM*2 и *COM*4 - *IRQ*3 или *IRQ*10. В принципе номер прерывания можно назначать в произвольных сочетаниях с базовым адресом (номером порта), но некоторые программы и драйверы (например, драйверы последовательной мыши) настроены только на стандартные сочетания. Каждому порту, нуждающемуся в аппаратном прерывании, обычно назначают отдельную линию, не совпадающую с линиями запроса прерываний других портов или устройств. Разделяемое использование линий прерывания адаптеров шин *ISA* проблематично. Прерывания необходимы для портов, к которым подключаются устройства ввода (мышь, дигитайзер), *UPS* и модемы. При подключении принтера или плоттера прерываниями пользуются только многозадачные ОС (и то не всегда), и этот дефицитный ресурс *PC* можно сэкономить. Также прерываниями обычно не пользуются и при связи двух компьютеров нуль-модемным кабелем.

*Использование канала *DMA* (для *UART* 16450 или 16550, расположенных на системной плате) - разрешение использования и номер канала *DMA*. Режим *DMA* при работе с COM-портами используют редко, поэтому в большинстве случаев каналы *DMA* порту не назначают.

Режим работы порта по умолчанию (2400 бит/с, 7 бит данных, 1 стоп-бит и контроль четности), заданный при инициализации порта во время *BIOS POST*, может изменяться в любой момент при настройке коммуникационных программ или командой *DOS MODE COMx*: с указанием параметров.

6.2.5. Использование COM-портов

Вопреки названию, COM-порты чаще всего используют для подключения манипуляторов (мышь, трекбол). В этом случае порт используется в режиме последовательного ввода, обеспечивая питание устройства от интерфейса. Мышь может подключаться к любому исправному порту, для согласования разъемов порта и мыши возможно применение переходника *DB*9*S*-*DB*25*P* или, наоборот,*DB*25*S*-*DB*9*P*. Для работы с мышью обязательно требуется использование линии прерывания, причем для порта COM1 - *IRQ*4, а для *COM*2 - *IRQ*3.

Следующим по популярности идет подключение внешних модемов для связи с удаленными компьютерами или выхода в глобальные сети. Модемы должны подключаться полным (9-проводным) кабелем *DTE*-*DCE*. Этот же кабель может использоваться и для согласования разъемов (по количеству контактов), возможно и применение переходников 9-25, предназначенных для мышей. Для работы коммуникационного ПО обычно требуется использование прерываний, но здесь, как правило, больше свободы выбора сочетаний номера (адреса) порта и номера линии прерывания. Если предполагается работа на скоростях 9600 бит/с и выше, то COM-порт должен быть реализован на микросхеме *UART* 16550*A* или совместимой с ней. Возможности работы с использованием *FIFO*-буферов и обмена по каналам *DMA* зависят от коммуникационного ПО.

Для связи двух компьютеров, удаленных друг от друга на небольшое расстояние, используют и непосредственное соединение их COM-портов нуль-модемным кабелем. Использование программ типа *Norton Commander* или *Interink MS-DOS* позволяет обмениваться файлами со скоростью передачи до 115,2 Кбит/с без использования аппаратных прерываний. Это же соединение может использоваться и сетевым пакетом *Lantastic*, предоставляющим более развитый сервис.

Подключение принтеров и плоттеров к COM-порту требует применения кабеля, соответствующего выбранному протоколу управления потоком: программному *XON*/*XOFF* или аппаратному *RTS*/*CTS*. Аппаратный протокол предпочтительнее, поскольку он не требует программной поддержки со стороны *PC*. Прерывания при выводе средствами *DOS* (командами *COPY* или *PRINT*) не используются.

COM-порт иногда используется и для подключения электронных ключей (*Security Devices*), предназначенных для защиты от нелицензированного использования программных продуктов. Эти устройства могут быть как "прозрачными", позволяя воспользоваться тем же портом и для подключения периферии, так и полностью занимающими порт.

COM-порт при наличии соответствующей программной поддержки позволяет превратить *PC* в терминал, эмулируя систему команд распространенных специализированных терминалов (*VT*-52, *VT*-100 и других). В принципе простейший терминал получается, если замкнуть друг на друга функции *BIOS* обслуживания COM-порта (*Int* 14*h*), функцию телетайпного вывода видеосервиса (*Int* 10*h*) и клавиатурный ввод (*Int* 16*h*). Однако такой терминал будет работать лишь на малых скоростях обмена (если, конечно, его делать не на *Pentium*), поскольку функции *BIOS* хоть и универсальны, но работают не самым быстрым образом.

Этим списком, конечно же, возможности использования COM-порта не исчерпываются. Интерфейс *RS*-232*C* широко распространен в различных периферийных устройствах и терминалах. Все они, при наличии должной программной поддержки, могут подключаться к *PC*. Кроме использования по прямому назначению, COM-порт может использоваться и как двунаправленный интерфейс, у которого имеется 3 программно-управляемых выходных линии и 4 программно-читаемых входных линии с двуполярными сигналами. Возможность их использования ограничивается только фантазией разработчика. Существует, например, схема однобитного широтно-импульсного преобразователя, позволяющего записывать звуковой сигнал на диск *PC*, используя входную линию COM-порта. Воспроизведение этой записи через обычный динамик обеспечивает разборчивость речи. Конечно, в настоящее время, когда звуковая карта стала почти обязательным устройством *PC*, это уже не впечатляет, но в свое время такое решение было довольно интересным

6.2.6. Низкоуровневое программирование *COM*-портов

Порт ***3F8h***.

Этот порт соответствует регистру передаваемых данных. Для передачи в порт 3*F*8*h* необходимо записать байт передаваемых данных.

После приема данных от внешнего устройства они могут быть прочитаны из этого порта. В зависимости от состояния бита управляющего слова, выводимого в управляющий регистр с адресом 3*F*8*h*, назначение порта 3*F*8*h* изменяться. Если этот бит равен 0, порт используется для записи передаваемых данных. Если же этот бит равен 1, порт используется для вывода значения младшего байта делителя частоты тактового генератора. Изменяя содержимое делите - ля, можно изменять скорость передачи данных.

Старший байт делителя записывается в порт 3*F*9*h*. Зависимость скорости передачи данных от значения делителя частоты приведены в таблице 3:

Таблица 3 — Зависимость скорости передачи данных от значения делителя частоты

Делитель	Скорость передачи в бодах.	Делитель	Скорость передчи в бодах.
1040	110	24	4800
768	150	12	9600
384	300	6	19200
192	600	3	38400
96	1200	2	57600
48	2400	1	115200

Порт ***3F9h***.

Порт используется как регистр управления прерываниями от асинхронного адаптера или (после вывода в порт 3*F*9*h* байта с установленным в 1 старшим битом) для вывода значения старшего байта делителя частоты тактового генератора. В режиме регистра управления прерываниями порт имеет следующий формат. Формат регистра приведен в таблице 4.

Таблица 4 — Регистр управления прерываниями

Бит	Значение
0	1 — разрешение прерывания при готовности принимаемых данных.
1	1 — разрешение прерывания после передачи байта (когда выходной буфер передачи пуст)
2	1 — разрешение прерывания по обнаружении состояния "*BREAK*" или ошибки.
3	1 — разрешение прерывания по изменению на разъёме *RS*-232-*C*.
4-7	Не используются, должны быть равны 0.

Порт ***3FAh***.

Регистр идентификации прерывания. По его содержимому программа может определить причину прерывания. Формат регистра приведён в таблице 5.

Таблица 5 — Регистр идентификации прерывания

Бит	Значение
0	1 — нет прерываний, ожидающих обслуживания.
1,2	00 — прерывание по линии состояния приёмника, возникает при переполнении приёмника, ошибка чётности или формата данных, или при состоянии "*BREAK*". Сбрасывается после чтения состояния линии и порта 3*FDh*. 01 — данные приняты и доступны для чтения. Сбрасывается после после чтения данных из порта 3*F*8*h*. 11 — Состояние модема. Устанавливается при изменении состояния входных линий *CTS*, *RI*, *DCD*, *DSR*.
3…7	Должны быть равны 0.

Порт ***3FBh***.
Управляющий регистр, доступен по записи и чтению. Его формат показан в таблице 6.

Таблица 6 — Формат управляющего регистра

Бит	Значение
0-1	Длинна слова в байтах: 00 - 5 бит; 01 — 6 бит; 10 — 7 бит; 11 — 8 бит.
2	Количество стоповых битов: 0 — 1 бит, 1 — 2 бита.
3…4	Чётность: 10 — контроль на чётность неиспользуется; 01 — контроль на нечётность; 11 — контроль на чётность.
5	Фиксация чётности. При установки этого бита бит чётности всегда принимает значение 0 (если биты 3…4 равны 11) или 1 (если биты 3…4 равны 01)
6	Установка перерыва. Вызывает вывод строки нулей в качестве сигнала "*BREAK*" для подключения устройства.
7	1 — порты 3*F*8*h* и 3*F*9*h* используется для загрузки делителя частоты тактового генератора; 0 — порты используются как обычно.

Порт ***3FCh***.
Регистр управления модемом.
Управляет состоянием выходных линий *DTR*, *RTS*, линий, специфических для модемов *OUT*1 и *OUT*2, для запуска диагностики при входе асинхронного адаптера, замкнутым на его выход. Формат порта приведён в таблице 7.

Таблица 7 — Формат регистра управления модемом

Бит	Значение
0	Линия *DTR*
1	Линия *RTS*.
2	Линия *OUT*1 (запасная)
3	Линия *OUT*2 (запасная)
4	Запуск диагностики при входе асинхронного адаптера, замкнутом на его выход.
5…7	Должно быть равно 0

Порт ***3FDh***.
Регистр состоянии линии. Значение зарядов регистра приведены в таблице 8.

Таблица 8 — Формат регистра состоянии линии

Бит	Значение
0	Данные получены и готовы для чтения, сбрасывается при чтении данных.
1	Ошибка переполнения. Был принят новый байт данных, а предыдущий ещё не был считан программой. Предыдущий байт потерян.
2	Ошибка чётности, сбрасывается после чтения состояния линии.
3	Ошибка синхронизации.
4	Обнаружен запрос на прерывание передачи "*BREAK*" — длинная строка нулей.
5	Регистр хранения передатчика пуст, в него можно записать новый байт для передачи.
6	Регистр сдвига передатчика пуст. Этот регистр получает данные из регистра хранения и преобразует их в последовательный вид для передачи.
7	Тайм-аут (устройство не связано с компьютером)

Порт ***3FEh***.
Регистр состояния модема. Значения битов указаны в таблице 9.

Таблица 9 — Формат регистра состояния модема

Бит	Значение
0	Линия *CTS* изменила состояние.
1	Линия *DSR* изменила состояние.
2	Линия *IR* изменила состояние.
3	Линия *DCD* изменила состояние.
4	Состояние линии *CTS*
5	Состояние линии *DSR*
6	Состояние линии *IR*.
7	Состояние линии *DCD*.

Приём и передача данных. Перед записью байта данных в регистр передатчика нужно убедиться, что регистр хранения передатчика свободен, то есть убедиться в том, что передача предыдущего символа завершена. Признаком свободы регистра передатчика является установленный в 1 бит 5 регистра состояния линии с адресом 3*FDh*.

Аналогично передачи данных перед вводом символа из порта приёмника 3*F*8*h* следует убедиться, что бит 0 порта 3*FDh* установлен в 1, то есть что символ принят из линии и находиться в буферном регистре приёмника.

6.3. Методические рекомендации к выполнению работы

6.3.1. Описание лабораторного макета

Макет для исследования работы *COM*-порта состоит из платы управления и дисплея. Внешний вид макета изображен на рис. 4. Для выполнения работы необходимо чтобы выход макета *RS*-232 был подключен к *COM*-порту компьютера и на вход питания должно быть подано 12 В. Выход 5 В предназначен для питания дополнительной переферии (в данной работе не используется).

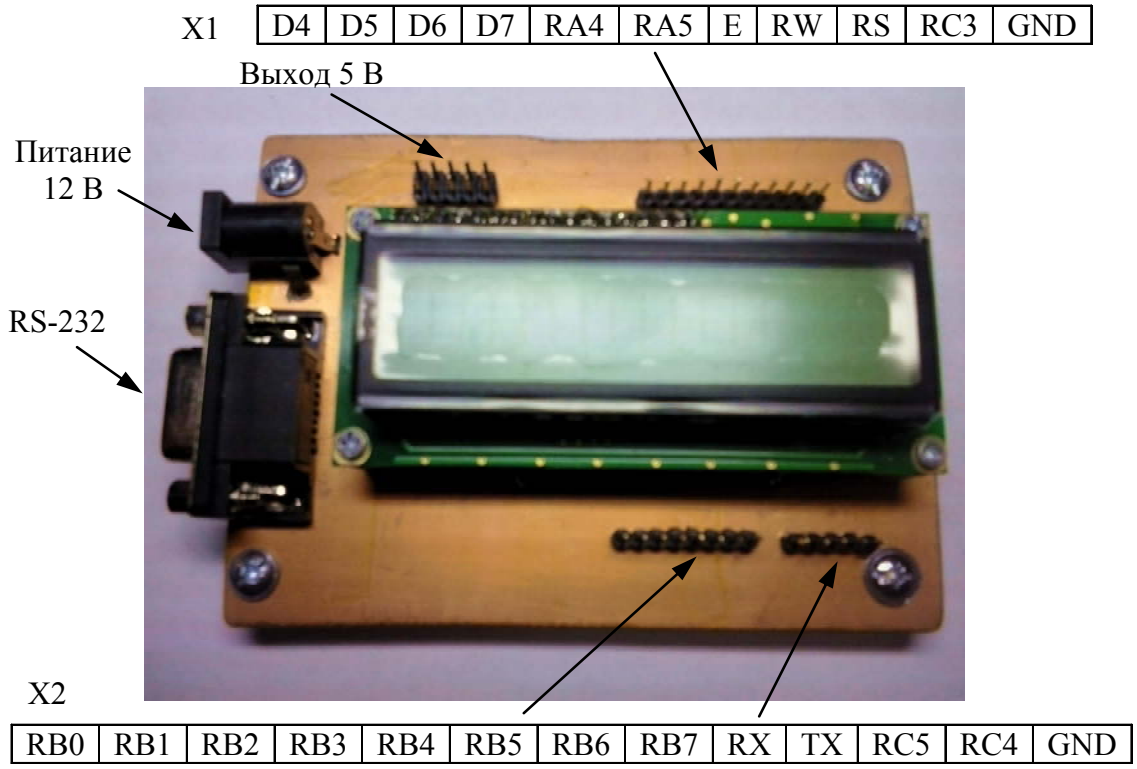

Рисунок 4 — Внешний вид лабораторного макета

Контакты разъёмов X1 *D*4…*D*7, а также *E*, *RW*, *RS* подключены к соответствующим выводам дисплея. А контакты *RA*4, *RA*5, *RC*3 подключены к соответствующим выводам контроллера.

Контакты разъёма X2 с *DB*0 по *DB*7 и *RC*5, *RC*4 подключены к соответствующим выводам контроллера. Контакты *RX* и *TX* подключены к выводам порта *RS*-232.

Для осуществления передачи данных необходимо настроить следующие параметры *COM*-порта компьютера:

— скорость передачи 9600 бод;

— количество передаваемых бит 8;

— проверка чётности отключена;

— стоп бит 1;

— *Handshaking* отключено.

Формат данных, в соответствии с которым осуществляется обмен, изображен на рисунке 5. На рисунке 5 *D*0…*D*7 — биты передаваемых данных.

Признак команды								Передаваемые данные							
*D*7	*D*6	*D*5	*D*4	*D*3	*D*2	*D*1	*D*0	*D*7	*D*6	*D*5	*D*4	*D*3	*D*2	*D*1	*D*0

Рисунок 5 — Формат передачи данных

Передача команд осуществляется в два этапа: сначало передаётся признак каоманды (символ %, @ или ~), затем передаётся команда. Перечень команд приведён в таблице 10.

Передача текстовых символов осуществляется в три этапа: сначало передаётся признак каоманды %, затем передаётся номер разряда или команда *p*, после чего передаётся сам символ.

Таблица 10 — Режимы передачи данных

Формат передачи данных	Результат выполнения
%*ND*	Запись в разряд с номером *N*
%*pD*	Вывод в следующий разряд
%*h*	Установить курсор в первый разряд
%*k*	Очистка дисплея
%*l*	Сдвиг курсора влево
%*r*	Сдвиг курсора вправо
%*m*	Включить мерцающий курсор
%*n*	Включить нижний курсор
%*o*	Выключить курсор
%*s*	Сдвиг дисплея вправо
@*N*	Чтение из разряда с номером *N*
~*d*	Эхо отключено
~*e*	Эхо включено

Примечание *N* — номер разряда на дисплее (1…9 и *a*…*g*); *D* — передаваемые данные; кодовая таблица символов *ASCII*.

6.3.2. Описание работы с *COM*-портом в среде *Borland C++ Builder*

Перед началом работы с *COM*-портом необходимо определить количество портов в системе и их имена. Также необходимо выбрать имя порта к которому подключен макет. Для этого на форму необходимо добавить три объекта:

— *ComboBox* (с именем *cports*) список где будут сохранены имена портов известные системе;

— *BitBtn* (с именем *refCom*) кнопка нажатие которой приводит к обновлению списка *cports* (происходит опрос системы на предмет наличия *COM*-портов);

— *SpeedButton* (с именем *connCom*) кнопка нажатие которой приводит к подключению/отключению к/от *COM*-порта.

При этом обработчик события *onClick* для кнопки *refCom* будетиметь следующий вид:

```
//---------------------------------------------------------------------------
void __fastcall TForm1::refComClick(TObject *Sender)
{
char buf[10*2048]; // Буфер для хранения ответа системы
int bufsize=10*2048; // Размер буфера
 QueryDosDevice(NULL, buf, bufsize) ; // API функция возвращающая перечисление переферийных устройств системы
 cports->Clear(); // Очищаем старые данные списка
 AnsiString ln="";
 int numi=0;
    for(int i=0;i<10*2048;i++) // Пробегаем по всему ответу системы
     {
     if(buf[i]=='\0') // Если конец строки
     {
     if(ln=="") break; // если пустая строка то перейти к следующей
     if(ln.SubString(1,3).UpperCase()=="COM") // Если три первых символа строки — COM значит предположительно найден COM-порт
     if(ln.SubString(4,1).ToIntDef(-1)!=-1) //Если после букв следует цифра то точно наден COM-порт
     {
     cports->AddItem(ln,0); // Добавляем найденный COM-порт к нашему списку
     cports->ItemIndex=0; // По умолчанию выбираем первый элемент списка

     }
     ln=""; // Инициализируем следующею строку ответа системы
     }    else ln=ln+buf[i]; // Если не конец строки то читаем символ из буфера
     }
}
//---------------------------------------------------------------------------
```

Для работы с *COM*-портом будем использовать специальный класс который расположен на методическом сервере в файлах «*Comm.h*» и «*Comm.cpp*» эти файлы необходимо скачать с сервера и поместить в папку проекта. Чтобы подключить этот класс к проекту необходимо в файле «*Unit*1.*cpp*» проекта добавить директиву «*#include* "*wake.cpp*"».

```
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
#include "wake.cpp"
```

При старте приложения необходимо создать экземпляр класса *TComm*.

```
//---------------------------------------------------------------------------
__fastcall TForm1::TForm1(TComponent* Owner)
      : TForm(Owner)
{
refComClick(NULL); // Обновляем список портов путем принудительного вызова события onClick кнопки refCom
com = new TComm(NULL); // Создаем экземпляр класса TComm
com->OnRxChar= RxCharEvent; // Назначить обработчик события приема символа
}
//---------------------------------------------------------------------------
```

Также необходимо создать обработчик события приема символа.

```
//---------------------------------------------------------------------------
void __fastcall TForm1::RxCharEvent(TObject* Sender, DWORD Count)
{
char buf[2048]; // Буфер приема
int lend; // Количество принятых байт
lend=com->Read(buf, Count) ; // Чтение принятых байт
// Здесь должна быть обработка принятых данных
}
//---------------------------------------------------------------------------
```

Это событие, а также указатель *com* надо прописать в файле «*Unit*1.*h*» проекта, чтобы они стали доступны другим элементам программы

```
public:          // User declarations
TComm *com; // Указатель на экземпляр класса TComm для работы с COM-портом
void __fastcall RxCharEvent(TObject* Sender, DWORD Count); // Декларируем обработчик
```

события приема байта из *COM*-порта.

Для передачи данных в *COM*-порт необходимо воспользоваться методом класса *TComm*.

```
char txUSARTbuf[2048];
    com->Write(txUSARTbuf, 5); // Передать 5 байт из буфера txUSARTbuf в COM-порт
```

Для подключения к *COM*-порту необходимо использовать следующий обработчик:

```
//---------------------------------------------------------------------------
void __fastcall TForm1::connComClick(TObject *Sender)
{
 if(connCom->Caption=="&Disconnect") //Если отключаем порт то
 {
 if(com->Connected) // Если порт подключен то сначала отключаем
{
com->Close();
}
 }
 else // Если подключаем порт
 {
 if(com->Connected) // Если порт подключен то сначала отключаем
{
com->Close();
}
com->DeviceName=cports->Text; // Устанавливаем имя порта согласно выбору пользователя
com->ReadTimeout=1000; // Таймаут чтения 1 с.
com->WriteTimeout=1000; // Таймаут записи 1 с.
com->ReadBufSize=2048; // Размер внутреннего буфера чтения
com->WriteBufSize=2048; // Размер внутреннего буфера записи
com->BaudRate=br9600; // Скорость порта
com->StopBits=sb10; // Количество стоповых бит — 1
com->DataBits=da8; //  Количество бит данных — 8
com->Parity=paNone; // Контроль четности — отключить
com->FlowControl=fcNone; // Управление рукопожатием — отключить
com->Open(); // Открываем порта для обмена информцией
```

```
 }
 if(com->Connected) // Если удалось открыть порт то меняем надписи на кнопке
 {connCom->Down=true;connCom->Caption="&Disconnect";}
 else
  {connCom->Down=false;connCom->Caption="&Connect";}
}
//---------------------------------------------------------------------------
```

6.4. Индивидуальные задания

Программа должна попеременно выводить на индикатора лабораторного макета фамилию, инициалы студента, и время указанное в таблице 11. Интервал индикации:

— ФИО — 2 секунды

— Время согласно таблицы 11 — 10 секунд.

Таблица 3.14 — Индивидуальное задание

M (последняя цифра зачетной книжки)	Индикация
Четная	Текущее время (в формате ЧЧ:ММ:СС)
Нечетная	Время до конца пары (в формате ЧЧ:ММ:СС)

6.5. Содержание отчета

Отчет должен содержать:

1. Титульный лист;
2. Цель работы;
3. Описание и алгоритм работы программы;
4. Текст программы и результаты разработки программы;
5. Результаты работы программы;
6. Выводы.

Перед защитой отчета студент должен продемонстрировать работу программы.

6.6. Контрольные вопросы

6.6.1. Что такое стандарт *RS*-232*C*?

6.6.2. Опишите синхронный режим передачи последовательных данных.

6.6.3. Опишите асинхронный режим передачи последовательных данных.

6.6.4. Какие преимущества и недостатки последовательного интерфейса передачи данных по сравнению с параллельным?

6.6.5. Каков размер минимальной посылки *COM*-порта(в битах)?

6.6.6. Как связана информационная и канальная скорость *COM*-пота?

6.6.7. Какова скорость общения компьютера с лабораторным макетом? Можно ли изменить эту скорость (и почему)?

6.6.8. Опишите последовательность работы с *COM*-портом из пользовательского приложения.

6.6.9. Какова максимальная длинна нуль-модемного кабеля?

6.6.10. Чем отличается аппаратная реализация интерфейсов *RS*-485 и *RS*-232*C*.