# 7 Лабораторная работа №7. Интерфейс RS-232C

**Цель работы:** изучить основные принципы работы последовательного интерфейса по протоколу RS-232С

## 7.1 Краткие теоретические сведения

Последовательный интерфейс для передачи данных использует одну сигнальную линию, по которой информационные биты передаются последовательно друг за другом. Отсюда происходит название интерфейса и порта: Serial Interface и Serial Port. Последовательная передача позволяет сократить количество сигнальных линий и увеличить дальность связи. В ряде последовательных интерфейсов применяется гальваническая развязка внешних сигналов от схемной земли устройства, что позволяет соединять устройства, находящиеся под разными потенциалами.

Последовательная передача данных может осуществляться в асинхронном или синхронном режимах. При асинхронной передаче каждому байту предшествует старт-бит, сигнализирующий приемнику о начале посылки, за которым следуют биты данных и, возможно, бит паритета (см. рисунок 7.1). Завершает посылку стоп-бит, гарантирующий паузу между посылками. Старт-бит следующего байта посылается в любой момент после стоп-бита, то есть между передачами возможны паузы произвольной длительности. Старт-бит, имеющий всегда строго определенное значение (логический 0), обеспечивает простой механизм синхронизации приемника по сигналу от передатчика. Подразумевается, что приемник и передатчик работают на одной скорости обмена. Очевидно, что при передаче 8 бит данных, одного контрольного и одного стоп-бита предельно допустимое рассогласование скоростей, при котором данные будут распознаны верно, не может превышать 5%. С учетом фазовых искажений и дискретности работы внутреннего счетчика синхронизации реально допустимо меньшее отклонение частот. С уменьшением коэффициента деления опорной частоты внутреннего генератора возрастает погрешность привязки стробов к середине битового интервала и требования к согласованности частот становятся более строгими. С увеличением частоты передачи возрастает влияние искажений фронтов на фазу принимаемого сигнала. Взаимодействие этих факторов приводит к повышению требований к согласованности частот приемника и передатчика с ростом частоты обмена.



Рисунок 7.1 – Формат передачи данных

Стартовый бит сигнализирует о начале передачи данных. Далее передаются биты данных, начиная с младшего. Если используется проверка четности, то после данных передается бит четности (паритет), который имеет такое значение, чтобы общее количество единиц (или нулей) в битах данных и паритета было четно или нечетно. В самом конце передаются один или два стоповых бита, завершающих передачу байта. Затем уровень линии передачи снова устанавливается в единицу до прихода следующего стартового бита. Передатчик и приемник должны иметь одинаковые настройки по количеству бит в байте, проверке четности, количеству стоповых битов и скорости передачи данных.

**7.2 Порядок выполнения работы**

7.2.1 Запустите программу Com\_io.exe, которая необходимо для получения и отправки пакетов данных по интерфейсу RS-232C.

7.2.2 Изучите основные возможности программы Com\_io.exe.

7.2.3 Запустите САПР QuartusII 9.0, создайте новый проект и подготовьте Text Editor для работы.

7.2.4 Опишите алгоритм передачи данных по последовательному интерфейсу RS-232C на поведенческом уровне языка VHDL с учетом вариантов заданий, которые указаны в таблице 7.2.

Таблица 7.2 – Варианты заданий

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № | Четность/  нечетность | Скорость  передачи  (бит/с) | Кол-во  битов  данных | Кол-во стоп-  битов | Передатчик | Приемник |
| 2 | Нечетность | 9600 | 7 | 1.5 | PLD | ПК |

7.2.5 Произведите компиляцию проекта с помощью команды Process/Start Compilation.

7.2.6 Назначьте номера контактов цифрового устройства в соответствии с требованиями, приведенными в таблице 7.5 и 7.6.

7.2.7 Опишите в Waveform Editor входные сигналы, а затем получите и исследуйте временные диаграммы работы приемника и передатчика   
RS- 232C.

Временные диаграммы работы приемника и передатчика   
RS- 232C представлены на рисунках 7.4 и 7.5.

7.2.8 Включите стенд и выберите режим Byte-Blaster→PLD.

7.2.9 Произведите загрузку готового проекта в ПЛИС с помощью программатора (Programmer).

7.2.10 Установите параметры программы Com\_io.exe в соответствии с рекомендациями, приведенными на рисунке 7.6.

7.2.11 Оцените правильность работы приемника и передатчика   
RS-232C.

**7.3 Результаты выполнения работы**

library ieee;

use ieee.std\_logic\_1164.all;

entity rs232 is

generic (

MAX\_STATE : integer := 34;

PDU\_SIZE : integer := 7;

MIN\_DATA\_STATE : integer := 2;

MAX\_DATA\_STATE : integer := 15;

LAST\_STOP\_BIT\_STATE : integer := 20;

MAX\_DIV : integer := 2; --512

MESSAGE\_LENGTH : integer := 10;

DATA\_LENGTH : integer := 7;

PARITY\_STATE : integer := 17

); --1024/2

port (

reset : in std\_logic; --сброс

clk : in std\_logic; --синхронизация

outbit : out bit;

test : out bit;

test1 : out bit

); -- выход

end rs232;

architecture outrs232\_arh of rs232 is

type arr is array (integer range 0 to MESSAGE\_LENGTH - 1) of bit\_vector(0 to (DATA\_LENGTH - 1));

signal data : bit\_vector (0 to (DATA\_LENGTH - 1));-- вектор данных

signal msg : arr; -- массив хранения данных

signal divided\_clk : std\_logic;

begin

msg(0) <= "1110100"; --t

msg(1) <= "1100101"; --e

msg(2) <= "1110011"; --s

msg(3) <= "1110100"; --t

msg(4) <= "1100000"; --

msg(5) <= "1110010"; --r

msg(6) <= "1110011"; --s

msg(7) <= "1110010"; --2

msg(8) <= "1110011"; --3

msg(9) <= "1110100"; --2

process (clk, reset)

variable div\_cnt : integer range 0 to MAX\_DIV;

begin

if reset = '0' then

div\_cnt := 0;

elsif (clk'event and clk = '1') then

div\_cnt := div\_cnt + 1;

divided\_clk <= '0';

test <= '0';

if (div\_cnt = MAX\_DIV) then

divided\_clk <= '1';

test <= '1';

div\_cnt := 0;

end if;

end if;

end process;

process (reset, divided\_clk)

variable state : integer range 0 to MAX\_STATE;

--variable data : std\_logic\_vector (0 to (DATA\_LENGTH - 1));

variable pdu\_cnt : integer range 0 to MESSAGE\_LENGTH;

begin

if (reset = '0') then

state := 0;

pdu\_cnt := 0;

else

if (divided\_clk'event and divided\_clk = '1') then

if (pdu\_cnt /= MESSAGE\_LENGTH) then

data <= msg(pdu\_cnt);

if (state / 2 = 0) then

outbit <= '0';

elsif (state >= MIN\_DATA\_STATE and state <= MAX\_DATA\_STATE) then

test1 <= '1';

outbit <= data(state / 2 - 1);

elsif (state > MAX\_DATA\_STATE and state <= PARITY\_STATE) then

test1 <= '0';

outbit <= not( data(0) xor data(1) xor data(2) xor data(3) xor data(4) xor data(5) xor data(6));

elsif (state > PARITY\_STATE and state <= LAST\_STOP\_BIT\_STATE) then

outbit <= '1';

test1 <= '1';

else

test1 <= '0';

outbit <= '1';

end if;

state := state + 1;

if (state = MAX\_STATE) then

pdu\_cnt := pdu\_cnt + 1;

state := 0;

end if;

end if;

end if;

end if;

end process;

end outrs232\_arh;

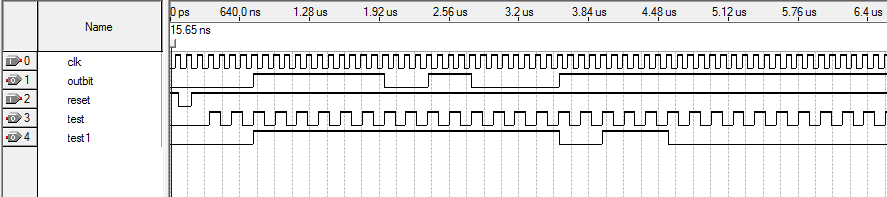


Рисунок 7.2 – Временные диаграммы передачи данных

**7.4 Особенности САПР Quartus II выявленные в ходе выполнения лабораторной работы**

В ходе лабораторной работе никаких особенностей САПР Quartus II выявлено не было.

**Выводы**

В ходе лабораторной работы была изучена передача данных с помощью интерфейса RS-232C. Были изучены принципы работы с последовательным портом в стенде PLD Emulator. Был описан передатчик сообщения с ПЛИС на ЭВМ по последовательном порту на поведенческом уровне языка описания аппаратуры VHDL.