Synthesis of a terminal device of a multiplex data exchange channel to FPGA

V I Zharkov¹, M N Petrov² and I Y Griskevich³

¹JBS Special Design Office “OMEGA”, Veliky Novgorod, Russian Federation
²Yaroslav-the-Wise Novgorod State University, Veliky Novgorod, Russian Federation
³BS "NTTs ELINS", Moscow, Russian Federation

E-mail: Mikhail.Petrov@novsu.ru

Abstract. The article presents a technology for developing a terminal device for a multiplexed serial interface for data exchange to FPGA. The development was carried out using the System Verilog hardware description language. To meet demanding requirements, presented to this interface in terms of transmission speed, as well as the duration of transmitted and received bits of information, the device includes a built-in signal duration analyzer, which can also be used to exclude the influence of a part of the impulse noise that “leaked” through the input high-pass filter. The presented device has passed the verification of the main operating modes in the ModelSim-Altera software environment.

1. Introduction

The purpose of this work is to develop a transceiver device based on FPGA, which performs the function of a terminal device as part of a multiplex serial data exchange interface (MCPI). For the implementation of such devices, three approaches are most widely used [1]:

- Synthesis of a device based on separate components, including separate functional blocks, for example, a decoder, an encryptor, etc.
- The second option involves the formation of a terminal device in the form of a module, implemented in one file, without splitting into separate functional modules.
- The third option involves the use of a massive multiplexer, which describes the operation of the device when receiving / transmitting information.

In all cases, the synthesis of the device is carried out on the basis of using the capabilities of one of the hardware description languages [2].

In this work, for the implementation of the terminal device, the first option is used, which, in contrast to the other two options, is simpler and more reliable. This is due to the fact that debugging the work of individual submodules is much easier than debugging an entire project, which greatly simplifies the search and elimination of possible errors and collisions. A feature of this work is that counters were used to detect the signal, and not shift registers, as in the version proposed in [3].

This approach also makes it possible to increase the accuracy of the received information, since here, for signal recognition, the developer can set the deviation value, within which the received information will be recognized as suitable. In addition, the advantage of this work in comparison with [3] is the avoidance of using a forty-bit register in the transmitting module of the device, which has a positive effect on the area occupied by the module.
It should be noted that the third option, which uses a massive multiplexer, was not considered for the development of this device, since the use of such a structure leads to a significant increase in the number of lines of code required to describe the device.

2. Main components of the terminal device

The main components of the developed terminal device include: a signal receiver (decoder), a signal transmitter (encoder), a top-level module (top module) that controls the operation of the receiver and transmitter.

To improve the accuracy of information reception, as well as the possibility of evaluating the validity of the received bit by its duration, the signal receiver was formed on the basis of three counters. The first counter counts the duration of one bit of information in clock cycles of the clk frequency generator (the number of clock cycles per bit, which is set in the code by the ONE_BIT_TIME parameter; in the example below, this counter is called cnt_to_50). The second (cnt_pol_bite1) and third (cnt_pol_bite2) counters count the number of levels of logical ones in the first and second half of the bit, respectively. The obtained values are compared with the minimum duration, which is considered valid (in the code it is indicated by the developer as GOOD_IMP). Thus, not only signal detection is possible, but also additional filtering of noise in the signal (if their amount does not exceed the permissible limits), which, together with the input HF filter, significantly reduces the number of errors. Of course, signal detection is also possible with the use of shift registers, but this complicates signal detection - in the worst case, the transmitted bit duration is minimal and the RF noise passed through the filter without changes. In addition, the use of high-capacity shift registers is also undesirable, since in this case they occupy a significant area, which can negatively affect performance. Therefore, the decision to build a signal detector on the counters can be considered acceptable. The fragment of the code responsible for detecting the bit of the transmitted word is given below.

```
always_ff @(posedge clk) begin
    if (state == WRITING) begin
        cnt_pol_bite1 <= #100 cnt_pol_bite2 <= #100;
    end
    if (state == COMMANDWORD || state == WRITE) begin
        cnt_to_50 <= ONE_BIT_TIME/100 * 10
        if (cnt_to_50 <= GOOD_IMP & & cnt_pol_bite1 <= ONE_BIT_TIME/100) det_cnt_bite <= 1'b0;
        if (cnt_to_50 <= GOOD_IMP & & cnt_pol_bite2 <= ONE_BIT_TIME/100) det_cnt_bite <= 1'b0;
    end
end
```

Figure 1. A fragment of the code responsible for detecting the received bit.

In turn, the transmitter of the terminal device is built on the basis of a counter, a register containing the word that needs to be transmitted, as well as a one-bit register, with the help of which half-bit meanders are formed. These half-bit square waves are needed to use a logical operation such as "exclusive or" (XOR), which greatly simplifies the structure of the transmitter. The code fragment responsible for the implementation of signal transmission to the line is shown in Figure 2.
Figure 2. A code fragment responsible for transferring information to the bus controller.

The above fragment shows a block responsible for unloading any type of words (response word or data word), and the sync pulse of the transmitted word, determined by the above-standing module, depending on the state of the terminal device (read state, that is, data word transmission, or the state of the response word and etc.). The sign of the word type is the i_type signal, and the i_en_transmit and i_block_transmiter signals also come from the top module and are responsible for enabling the transmitter. Cnt_bit_transmit and cnt_rd are the counters of the transmitted bit and the clock in the transmitted word, respectively.

Figure 4 shows a simulation of receiving the control word (shown in Figure 3), from which you can see that the word was received and recognized completely (the counters read the level transitions on the i_data channel and fill the register containing the o_com_wrd command word). In the state column, the state of the receiver is highlighted.

Figure 3. A fragment of the code responsible for setting the command word in the simulation.

Below is a simulation of the terminal device unloading the response word in response to the received data packet, from which it can be seen that the transmitter is working adequately.
So, the two main nodes of the terminal device have already been confirmed to work. Let's move on to the development of the state machine (Figure 6). It should be clarified that the developed device does not execute the entire range of commands supported by MCPI interface. This is due to the fact...
that the terms of reference indicate a reduced set of commands that will be supported by a computer connected to the interface.

Figure 6 shows that to switch states, various flags are used, which serve as indicators of the end of reception or transmission of a word/or a packet of words. We also note that to “speed up” switching it was possible to use the “always_comb” block with blocking assignments, but since the developed scheme is designed to use the frequency clk ≥10 MHz, reworking this block does not make sense, since the transmission rate and time intervals pauses are strictly within tolerance.

3. Conclusion
In this paper, the implementation of the main nodes of the terminal device for the multiplexed data exchange channel is presented, which make it possible to implement a reliable and relatively simple device that provides an opportunity for detecting input signals and sorting them by duration. In addition, this device contains a transmitter.

The main program blocks proposed in the work have been verified, which proves their adequate and stable operation.
References

[1] GOST P 52070-2003. Trunk serial interface of a system of electronic modules

[2] Sutherland S, Davidmann S and Flake P 2006 System Verilog for Design, 2-nd ed. (Springer Science + Business Media, LCC) p 408

[3] Daineko D Development of the MIL-STD 1553B protocol controller based on FPGA. Part 2
Available at: https://www.elibrary.ru/item.asp?id=21078868