Key debounce experiment in Vivado

Technical Email: alinx@aithtech.com Sales Email: rachel.zhou@alinx.com

1 Document Introduction

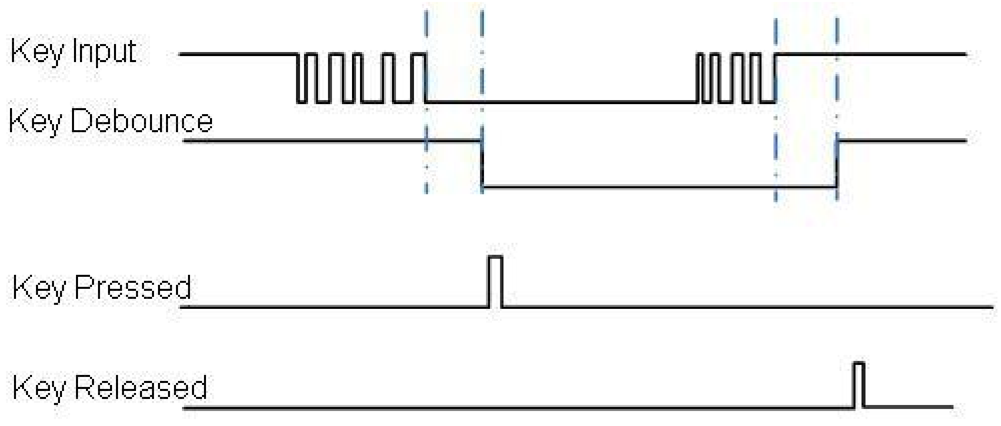
This article mainly explains the principle of key debounce and program writing. The program realizes the number plus 1 after key press one time and is displayed in led, compiled and debugged by vivado software.

2 Experiment Environment

- Windows 7 SPI 64 bit
- vivado (vivado2017.4)
- ALINX Brand FPGA Development Board (AXKU040 FPGA Development Board)

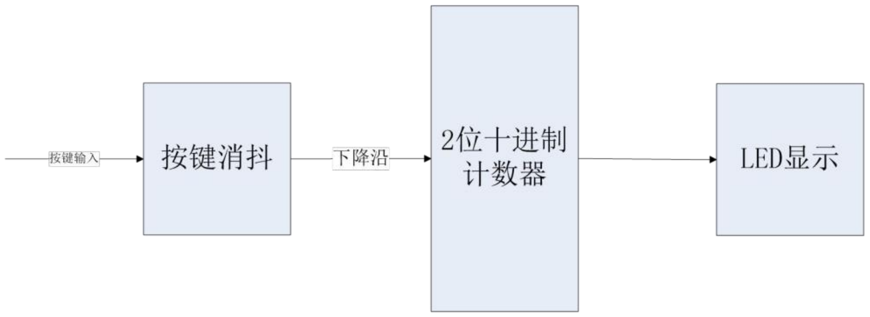
3 Experiment Principle

This experiment is mainly to debounce through FPGA timing. In the experiment, a counter is designed. When the key input changes, the timer is cleared. Otherwise, it is accumulated until it is added to a predetermined value (for example, 10ms). It is considered that the button is stable and the button value is output, so that the button value without jitter is obtained. Since the detection of the falling edge or rising edge of the button is required in many places, the button debounce module directly integrates the functions of rising edge and falling edge detection.



4 Programming

As shown in the figure below, after the key is debounced, the decimal counter is incremented by 1 when the key is pressed, and displayed after being decoded by the digital tube.



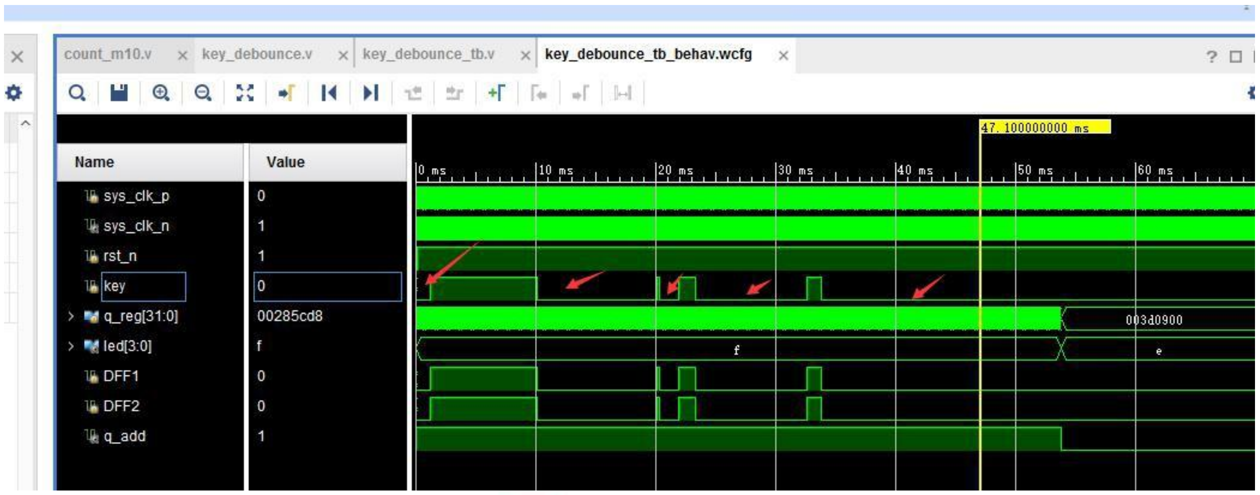
The principle of the key debounce part has been mentioned in the previous section. The code of the key debounce part is very refined, which is slightly confusing to read. It is recommended to read the code in combination with the simulation waveform. The simulation file “key_debounce_tb.v” is also provided in the “src” folder under the provided routine file.

You can simulate the change of the signal in the code by adding a simulation file.

Signal Name	Direction	Description
clk	in	Clock input
rst_n	in	Asynchronous reset input, low reset
button_in	in	Key input
button_posedge	out	After the debounce, the rising edge of the key, high effective, 1 clock cycle
button_negedge	out	After debounce, press the rising edge, high effective, 1 clock cycle
button_out	out	key output after debounce

Key debounce module (ax_debounce) port

The LED display section is not explained in this chapter.In the routine, the button debounce module "ax_debounce" module does some explanation. The module passes two levels of D flip-flops to register the key value, and only outputs the key value when the key value is stable. We can see that there is an "assign a_reset=(DFF1 ^ DFF2)" in the assign assignment statement. Anyone who has learned digital circuits should know that "^" is an XOR operator. The result of the same operation on both sides of the operator is 0, and the result of the different operation is 1. In the program, the values of the DFF1 and DFF2 comparison operations are assigned to "a_reset" by "assign", indicating whether the values of the two-stage registers before and after the comparison of the latched key values are the same, only the values of the two-stage registers are the same, that is, the a_reset A value of 0 indicates that the currently latched key has not changed. When the counter is added to "TIMER_MAX_VAL", it means that the latched key value is stable and can be output. Also in the module we can see the "{ }" symbol, to be noted that this is not the braces, which represents a bit concatenation operator which functions within the operator two, or multi-bit signal spliced together, Please refer to the routine for specific usage.

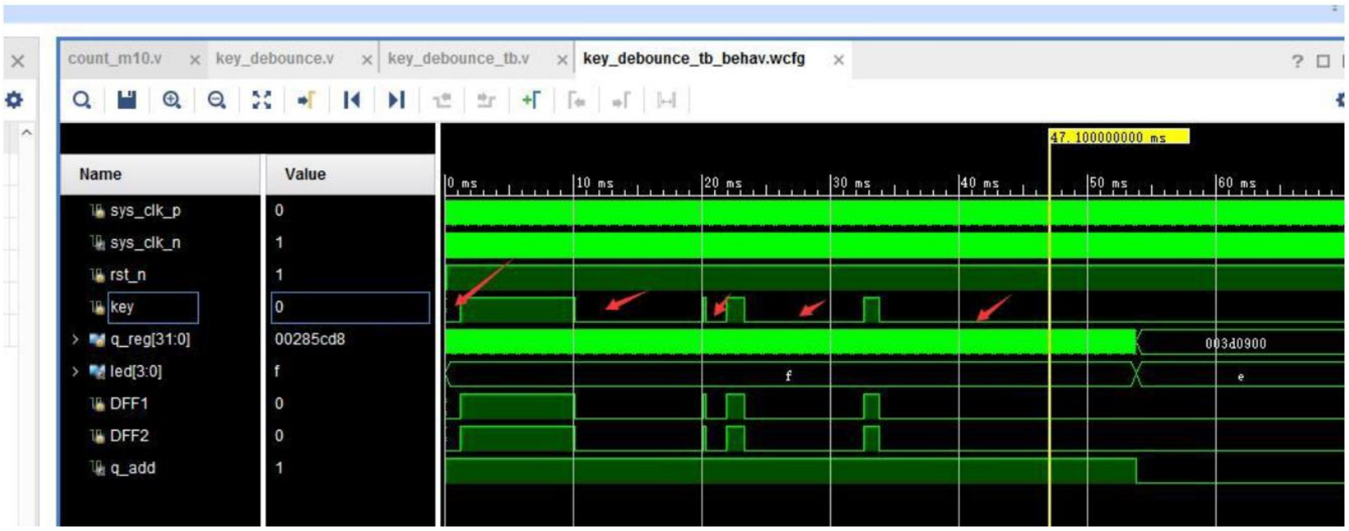


Finally, the program needs to explain the two output signals "button_posedge" and "button_negedge". This is a commonly used method for collecting rising and falling edges. The RTL view described is as follows:

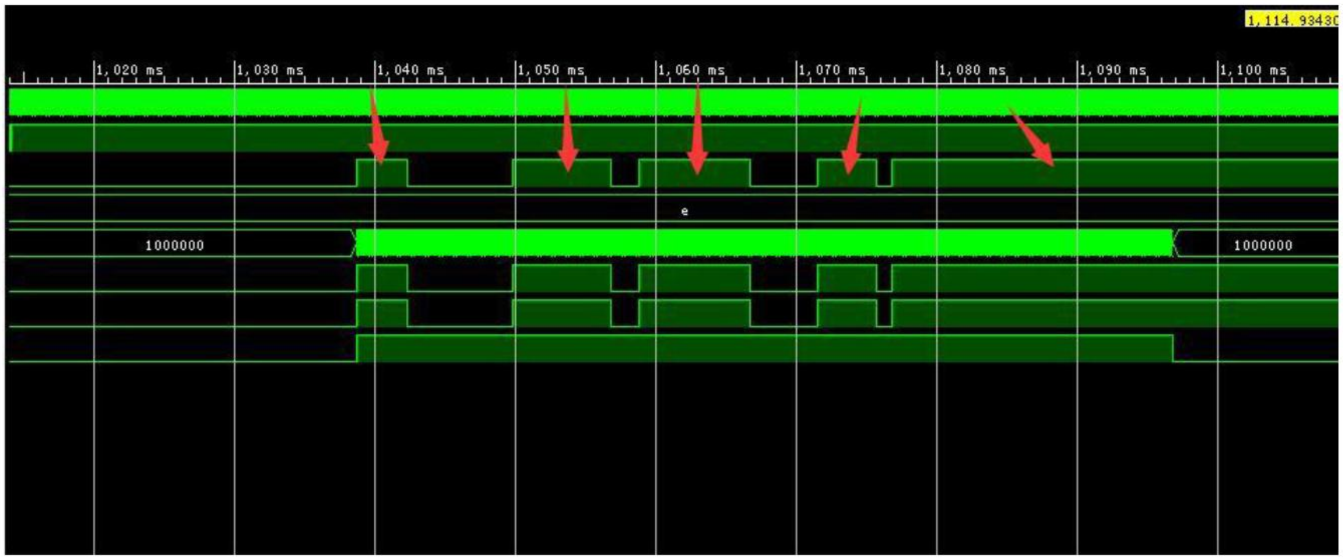
Of course, there are other methods for describing the edge detection circuit, but the basic principle is to first assign the signal to be detected as input non-blocking to a custom register in the logic timing circuit. It is judged whether it is a rising edge or a falling edge by judging the values of the two-stage registers before and after. The change from 0 ->1 is a rising edge, and the change from 1 -> 0 is a falling edge;

5 Simulation

Here we have added a serial port received stimulus “key_debounce_tb.v” file, used to simulate the input of the button key. The result of the simulation is as follows. We see that the button has been pressed a total of 5 times, but since the low level hold time of the previous 4 key presses is less than 10 ms, the four presses are judged to be jitter by the program. Only when the 5th key press time is greater than 10ms, it is judged that the button is pressed, and the state of the LED lamp changes.



After the button is released, it will also judge whether the high-level holding time is greater than 10ms, because the previous four high levels are lower than 10ms, the program uses them as the button jitter, only the fifth high level remains. When the time is longer than 10ms, the signal of “q_add” will change.



6 Experimental Result

After the development board is powered on, download the program, press the "KEY2" button, you can see that the 4 LEDs will change, corresponding to the binary data, press the button once to increase by one. If you do not debounce, it is impossible to achieve the increase by one.