

## Key detection experiment in Vivado

Technical Email: [alinx@aithtech.com](mailto:alinx@aithtech.com)

Sales Email: [rachel.zhou@alinx.com](mailto:rachel.zhou@alinx.com)

---

### 1 Document Introduction

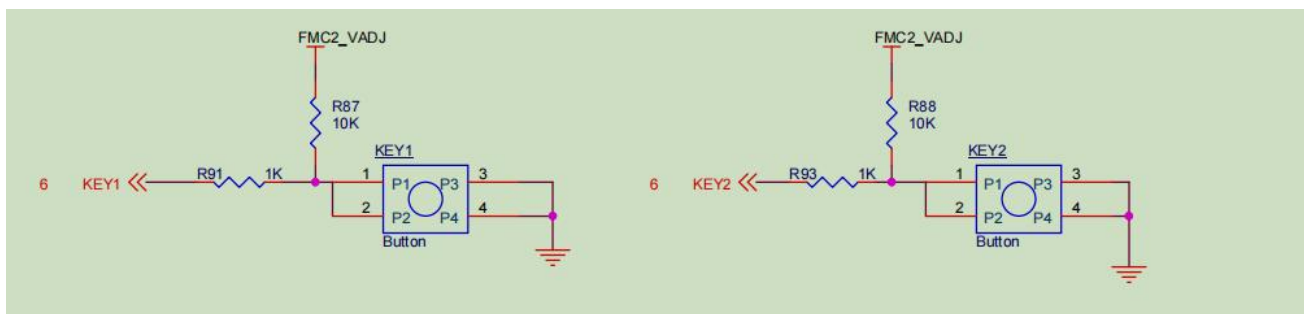
Through the button detection experiment, check whether the key function of the development board is normal, understand the specific relationship between the hardware description language and the FPGA, and learn the use of Vivado RTL ANALYSIS.

### 2 Experiment Environment

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

### 3 Experiment Principle

#### 3.1 Key Hardware Circuit

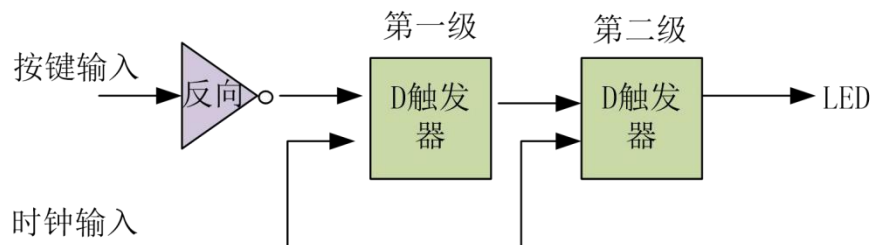


Keys Hardware Circuit on AXKU040 FPGA Development Board

As can be seen from the circuit, the key of the circuit is high when it is released, and low when it is pressed.

#### 3.2 Programming

The program looks at the connection between the hardware description language and the FPGA hardware through a simple hardware description language. First, we pass the key input through a non-gate and then pass through two sets of D flip-flops. The signal that passes through the D flip-flop is latched on the rising edge of the D flip-flop clock input and then sent to the output.



Before the hardware description language coding, we have completed the hardware construction, which is a normal development process. With hardware design ideas, whether through drawing or Verilog HDL, VHDL can complete the design, according to the design of complex programs and familiar procedures for a language to select tools.

## 4 Project Analysis

- (1) First, Create the test project of the key, add the verilog testing code, and complete the process of compiling and assigning pins

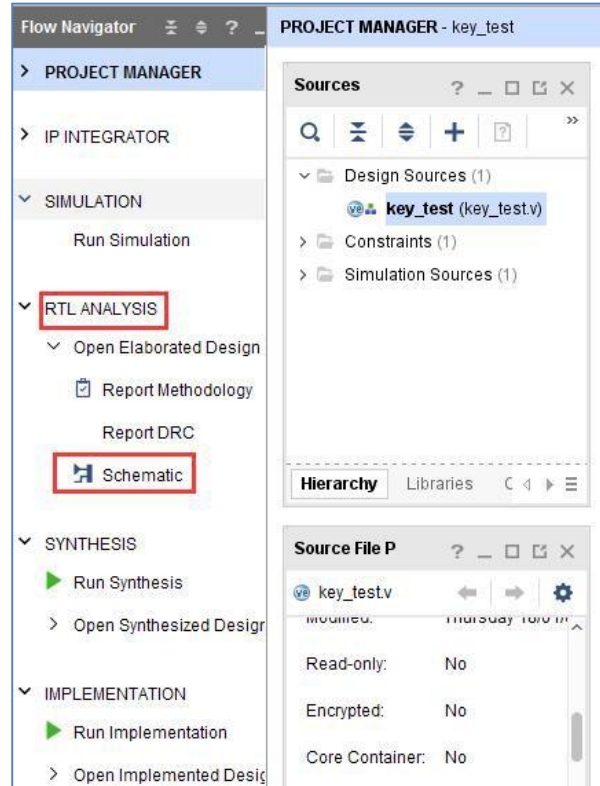
The screenshot displays the ALINX IDE interface. On the left, the **Flow Navigator** shows a tree view with categories like PROJECT MANAGER, IP INTEGRATOR, SIMULATION, RTL ANALYSIS, SYNTHESIS, and IMPLEMENTATION. The **PROJECT MANAGER - key\_test** window is open, showing a **Hierarchy** view with a tree structure: Design Sources (1) containing key\_test (key\_test), which includes Constraints (1) and Simulation Sources (1). Below this, a **Sour** window shows the **key\_test.v** file with properties: Enabled, Location: D:/demo, Type: Verilog, Library: xil\_defa, Size: 3.4 KB, and Modified: Today a. The main editor on the right shows the Verilog code for **key\_test.v**, which includes a timescale, module definition, input/output declarations, register declarations, and logic for an LED output.

```

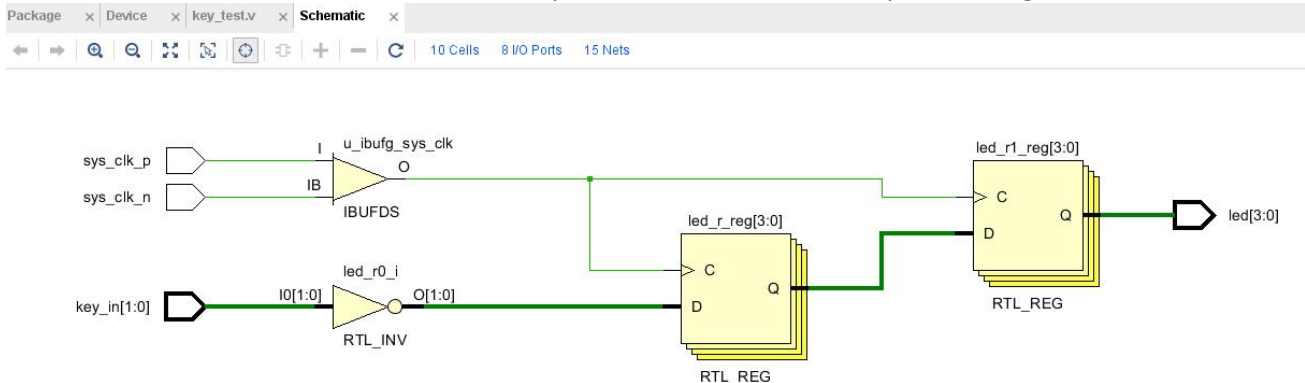
29
30 `timescale 1ns / 1ps
31 module key_test
32 (
33     input          sys_clk_p,      // Di
34     input          sys_clk_n,
35     input [1:0]    key_in,        //input
36     output[3:0]    led            //LED dis
37 );
38 reg[3:0]          led_r;          //d
39 reg[3:0]          led_r1;         //d
40 //=====
41 //Differentia system clock to single end cl
42 //=====
43 wire             sys_clk;
44 IBUFGDS #
45 (
46     .DIFF_TERM ("FALSE"),
47     .IBUF_LOW_PWR ("FALSE")
48 )
49 u_ibufg_sys_clk
50 (
51     .I (sys_clk_p),
52     .IB (sys_clk_n),
53     .O (sys_clk )
54 );
55 always@(posedge sys_clk)
56 begin
57     led_r <= {2{key_in}};        //first stage
58 end

```

(2) We can view the design using the “RTL ANALYSIS” tool



- (3) By analyzing the RTL graph, it can be seen that the first-stage D flip-flop is input after being inverted, and the second-level direct input is consistent with the expected design.



## 5 Experimental result

After the Bit program is downloaded to the development board, the AXKU040 FPGA development board, "LED1", "LED2", "LED3", "LED4" are all OFF.

Press "KEY1", Then "LED1" and "LED3" are all on, "LED2" and "LED4" are all off.

Press "KEY2", Then "LED2" and "LED4" are all on, "LED1" and "LED3" are all off.

## 6 Appendix

key\_test.v(verilog Code)

```
`timescale 1ns / 1ps

module key_test
(
    input          sys_clk_p,          // Differentia system clock 200Mhz input on board
    input          sys_clk_n,
    input [1:0]    key_in,             //input four key signal,when the keydown,the value is 0
    output [3:0]   led                 //LED display ,when the siganl high,LED lighten
);
    reg [3:0]      led_r;               //define the first stage register ,generate four
D Flip-flop
    reg [3:0]      led_r1;             //define the second stage register ,generate four
D Flip-flop
    //=====
    //Differentia system clock to single end clock
    //=====
    wire          sys_clk;
    IBUFGDS #
    (
        .DIFF_TERM    ("FALSE"),
        .IBUF_LOW_PWR ("FALSE")
    )
    u_ibufg_sys_clk
    (
        .I  (sys_clk_p),
        .IB (sys_clk_n),
        .O  (sys_clk )
    );
    always@(posedge sys_clk)
    begin
        led_r <= {2{~key_in}};         //first stage latched data
    end

    always@(posedge sys_clk)
    begin
        led_r1 <= led_r;               //second stage latched data
    end

    assign led = led_r1;
endmodule
```