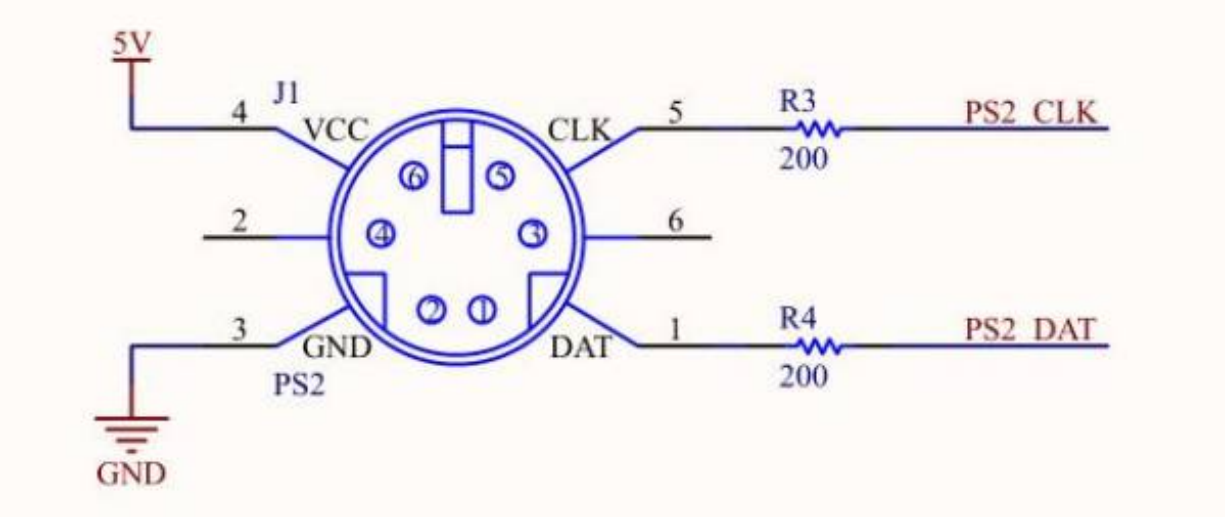
PS2 keyboard driver based on STEP FPGA

In this section, we will use FPGA to drive the keyboard peripherals of the PS2 interface on the backplane.

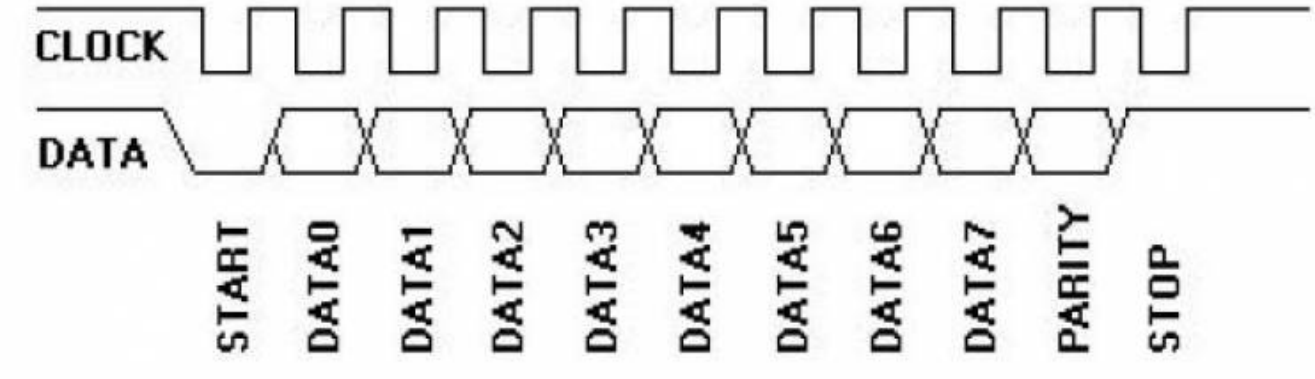
====Hardware description====

Our STEP-BaseBoard board integrates the PS2 keyboard interface, which can be used to connect the PS2 keyboard or PS2 mouse to complete the corresponding design. Next, let's understand the hardware connection of the PS2 interface and the driving method of the PS2 keyboard.
PS2 interface connection is very simple, only need to connect 4 wires:

- No. 4 pin VCC is connected to the power supply, generally 5V power supply, after testing, 3.3V can also be used
- No. 3 pin GND can be grounded
- The clock line of pin 5 and the data line of pin 1 are two bidirectional signal lines
- Pin 2 and Pin 6 are reserved pins and do not need to be connected



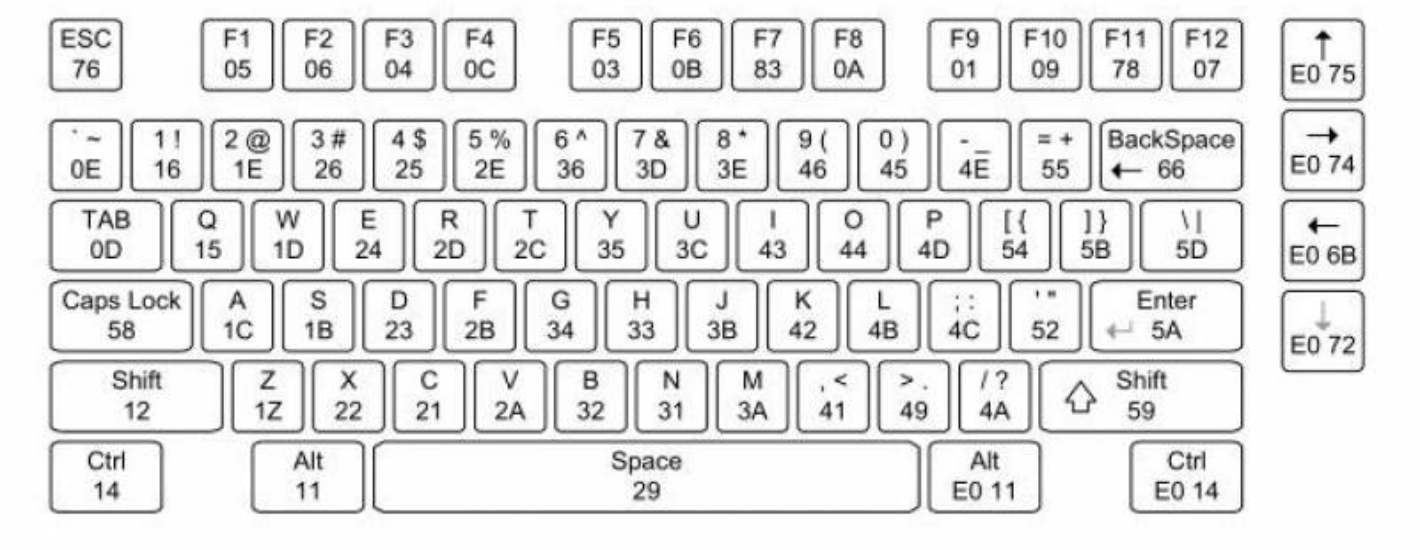
When there is a key press or operation on the PS2 keyboard, the keyboard will send a signal to the host. The timing of the clock signal and data signal of the PS2 interface is as shown in the figure below:



FPGA or host receives the data sent back by the keyboard, and the keyboard is determined by the keyboard coding rules In the current operation, there are two different types of scan codes: make code and break code. When a key is pressed or held down continuously, the keyboard will send the pass code of the key to the host; and when a key is released, the keyboard will send the break code of the key to the host.
According to the different scan codes of keyboard keys, the keys can be divided into the following categories:

- For the first type of keys, the pass code is 1 byte, and the break code is in the form of 0xF0+ pass code. For example, the A key has a pass code of 0x1C and a broken code of 0xF0 0x1C.
- For the second type of keys, the pass code is in the form of 2 bytes 0xE0+0xXX, and the broken code is in the form of 0xE0+0xF0+0xXX. For example, the right ctrl key, the pass code is 0xE0 0x14, and the break code is 0xE0 0xF0 0x14.
- There are two special keys of the third type, the print screen key-through code is 0xE0 0x12 0xE0 0x7C, the break code is 0xE0 0xF0 0x7C 0xE0 0xF0 0x12; the pause key-through code is 0x E1 0x14 0x77 0xE1 0xF0 0x14 0xF0 0x77, and the break code is empty.

The scan code of the combination key is sent in the order in which the keys occur, such as pressing the left SHIFT+A key in the following order: 1 press the left SHIFT key, 2 press the A key, 3 release the A key, 4 release the left SHIFT key, then the computer The string of data received is 0x12 0x1C 0xF0 0x1C 0xF0 0x12.
In the driver program design, different keys are processed differently according to this classification. The current simple program only supports the operation of the first type of keys.
The codes of different keys in the keyboard are as follows:



====Verilog code====

```
// ---------------------------------------------- --------------------
// >>>>>>>>>>>>>>>>>>>>>>>> COPYRIGHT NOTICE < <<<<<<<<<<<<<<<<<<<<<<<
// ------------------------ ----------------------------------------------
// Module: Keyboard_PS2
/ /
// Author: Step
//
// Description: PS2 keyboard driver
//
// Web: www.stepfapga.com
//
// -------------------- ----------------------------------------------
// Code Revision History:
// ---------------------------------------- ------------------------
// Version: |Mod. Date: |Changes Made:
// V1.0 |2016/04/20 |Initial ver
// ---------------------------------------------- --------------------
module Keyboard_PS2
 (
input                                   clk_in ,                                //System clock
input                                   rst_n_in ,                      //System reset, low ef
fective
input                                   key_clk ,                       //PS2 keyboard clock i
nput
input                                   key_data ,                      //PS2 keyboard data in
put
output  reg                             key_state ,                     //Keyboard pressed sta
te, press is 1, and release is 0
output  reg             [ 7 : 0 ]       key_ascii                        // ASCII code corresp
onding to key value
) ;

/*
This module is a simple program for FPGA to drive the PS2 keyboard. It can only support single
-key pressing of the first type of keys in the keyboard, and does not support simultaneous pre
ssing of multiple keys
*/

reg             key_clk_r0 =  1'b1 , key_clk_r1 =  1'b1 ;
reg             key_data_r0 =  1'b1 , key_data_r1 =  1'b1 ;
//Latch the keyboard clock data signal with delay
always  @  ( posedge clk_in or  negedge rst_n_in )  begin
        if ( ! rst_n_in )  begin
                key_clk_r0 <=  1'b1 ;
                key_clk_r1 <=  1'b1 ;
                key_data_r0 <=  1'b1 ;
                key_data_r1 <=  1'b1 ;
        end  else  begin
                key_clk_r0 <= key_clk ;
                key_clk_r1 <= key_clk_r0 ;
                key_data_r0 <= key_data ;
                key_data_r1 <= key_data_r0 ;
```

```
        end
end

//Keyboard clock signal falling edge detection
wire    key_clk_neg = key_clk_r1 &  ( ~ key_clk_r0 ) ;

reg                             [ 3 : 0 ]       cnt ;
reg                             [ 7 : 0 ]       temp_data ;
//Read data according to the falling edge of the keyboard clock signal. For details, refer to
 the PS2 keyboard data transmission format and timing
always  @  ( posedge clk_in or  negedge rst_n_in )  begin
        if ( ! rst_n_in )  begin
                cnt <=  4'd0 ;
                temp_data <=  8'd0 ;
        end  else  if ( key_clk_neg )  begin
                if (CNT > =  4'd10 ) CNT <=  4'd0 ;
                the else CNT <= CNT +  1'b1 ;
                Case  ( CNT )
                        4'd0 :  ;       // start bit
                        4'd1 : TEMP_DATA [ 0 ]  <= key_data_r1 ;   //Data bit bit0
                        4'd2 : temp_data [ 1 ]  <= key_data_r1 ;   //Data bit bit1
                        4'd3 : temp_data [ 2 ]  <= key_data_r1 ;  //Data bit bit2
                        4'd4 : temp_data [ 3 ]  <= key_data_r1 ;   //Data bit bit3
                        4'd5 : temp_data [ 4 ]  <= key_data_r1 ;   //Data bit bit4
                        4'd6 : temp_data [ 5 ]  <= key_data_r1 ;   //Data bit bit5
                        4'd7 : temp_data [ 6 ]  <= key_data_r1 ;   //Data bit bit6
                        4'd8 : temp_data [ 7 ]  <= key_data_r1 ;  // Data bit bit7
                        4'd9 :  ;       // parity bit
                        4'd10 :;        // End bit
                        default :  ;
                ENDCASE
        End
End

reg                                             key_break =  1'b0 ;
reg                             [ 7 : 0 ]       key_byte =  1'b0 ;
//Determine whether the button is currently pressed or released according to the pass code and
the break code
always  @  ( posedge clk_in or  negedge rst_n_in )  begin
        if ( ! rst_n_in )  begin
                key_break <=  1'b0 ;
                key_state <=  1'b0 ;
                key_byte <=  1'b0 ;
        end  else  if ( cnt== 4'd10  && key_clk_neg )  begin
                if ( temp_data ==  8'hf0 ) key_break <=  1'b1 ;         //Receive a segment co
de (8'hf0) to indicate that the button is released, and set the code break mark to 1
                else  if ( ! key_break )  begin         //When the break code is marked as 0,
 it means that the current data is pressed data, output the key value and set the pressed mark
ing to 1
                        key_state <=  1'b1 ;
                        key_byte <= temp_data ;
                end  else  begin        //When When the code break
                        flag is 1, it indicates that the current data is the release data, and
```

```
the code break flag and the press flag are cleared key_state <=  1'b0 ;
                        key_break <=  1'b0;
                end
        end
end

// Convert the effective key value returned by the keyboard to the ASCII code value correspond
ing to the key letter
always  @  ( key_byte )  begin
        case  ( key_byte )     //translate key_byte to key_ascii
                8'h15 : key_ascii =  "Q" ; //8'h51; //Q
                8'h1d : key_ascii =  "W" ; //8'h57; //W
                8'h24 : key_ascii =  "E" ; //8'h45; //E
                8'h2d : key_ascii =  "R" ; //8'h52; //R
                8'h2c :key_ascii =  "T"; //8'h54; //T
                8'h35 : key_ascii =  "Y" ; //8'h59; //Y
                8'h3c : key_ascii =  "U" ; //8'h55; //U
                8' h43 : key_ascii =  "I" ; //8'h49; //I
                8'h44 : key_ascii =  "O" ; //8'h4f; //O
                8'h4d : key_ascii =  "P" ; //8' h50; //P
                8'h1c : key_ascii =  "A" ; //8'h41;//A
                8'h1b : key_ascii=  "S" ; //8'h53; //S
                8'h23 : key_ascii =  "D" ; //8'h44; //D
                8'h2b : key_ascii =  "F" ; //8'h46; / /F
                8'h34 : key_ascii =  "G" ; //8'h47; //G
                8'h33 : key_ascii =  "H" ; //8'h48; //H
                8'h3b : key_ascii =  "J" ; //8'h4a; //J
                8'h42 : key_ascii =  "K"; //8'h4b; //K
                8'h4b: key_ascii =  "L" ; //8'h4c; //L
                8'h1a : key_ascii =  "Z" ; //8'h5a; //Z
                8'h22 : key_ascii =  "X" ; //8'h58 ; //X
                8'h21 : key_ascii =  "C" ; //8'h43; //C
                8'h2a : key_ascii =  "V" ; //8'h56; //V
                8'h32 : key_ascii =  "B " ; //8'h42; //B
                8'h31 :key_ascii =  "N" ;8'h4e //; // N
                8'h3a : key_ascii =  "M" ; // 8'h4d; M //
                default :  ;
        ENDCASE
End

endmodule
```

====Summary====

This section mainly explains the PS2 interface circuit, PS2 keyboard coding rules, and the simple method of using FPGA to drive the PS2 keyboard. You need to master it while creating your own project, and generate the FPGA configuration file loading test through the entire design process.
If you are not familiar with the use of Diamond software, please refer to here: Use of Diamond .

====Related Information====

Use STEP-MXO2 second generation of PS2 keyboard driver: download link will be updated, the subsequent use of the STEP-MAXIO PS2 keyboard driver: subsequent download link will be updated