Lab Assignment 2

Interfacing Keyboard with FPGA Board

Masoud Nouripayam

How does the keyboard send data?

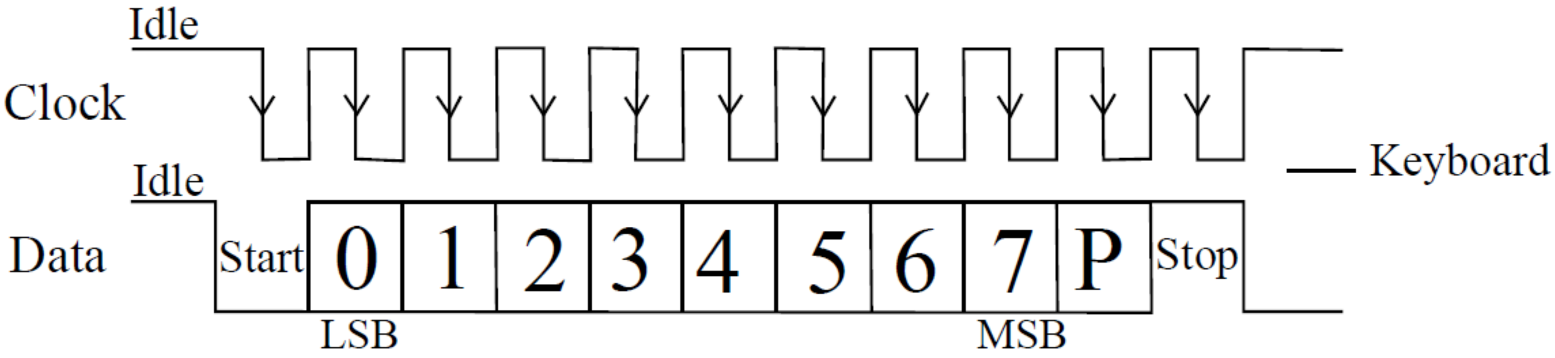
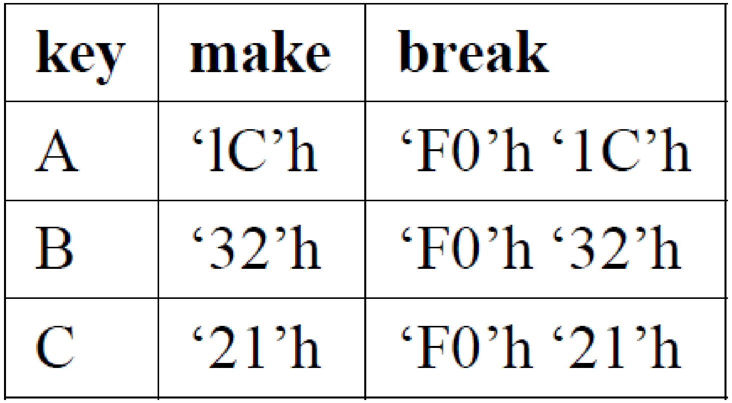


Figure 1: Key Board interface waveform

- **Old PS/2 Keyboard sends data serially**
- **Main code of each key is gathered in 8 bits (In addition to heading and tailing bits)**
- **Clock frequency is around 10-30 KHz**
- **Sampling edge is the falling edge**

What happens when you press/release a key?

- **Key pressed = make code is generated.**
- **Key released = break code is generated.**



key	make	break
A	‘1C’h	‘F0’h ‘1C’h
B	‘32’h	‘F0’h ‘32’h
C	‘21’h	‘F0’h ‘21’h

Example of how make/break codes work

Take for example the 'A' Key

When the key is pressed:

“1C” is sent serially through serial data line

**If you are still holding it down, another “1C” will be sent.
This keeps occurring until the 'A' key has been released**

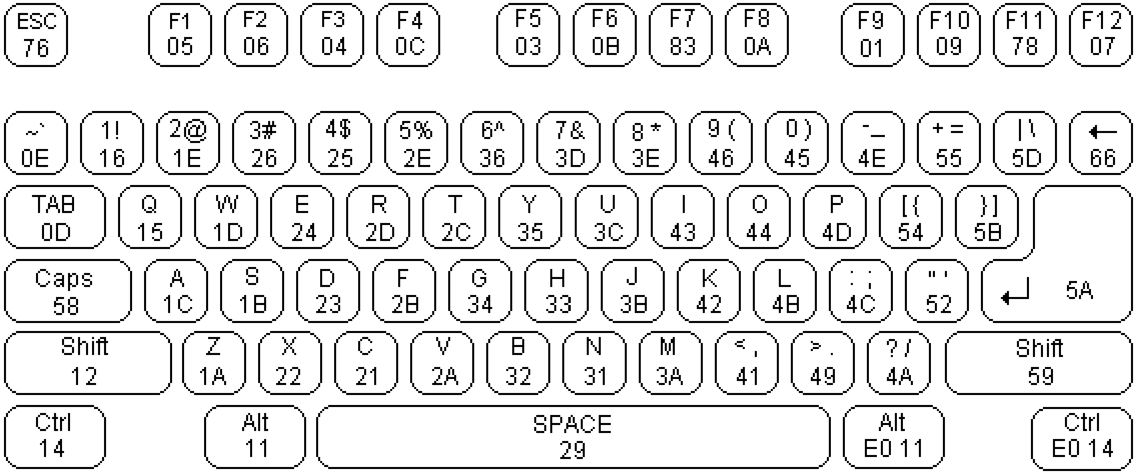
when the pressed key is released:

The keyboard will send “F0” (hex) and then “1C”

**For more info see:
https://retired.beyondlogic.org/keyboard/keybrd.htm**

Scan codes

- **Each key is assigned a unique scan code.**



ESC 76 | F1 05 | F2 06 | F3 04 | F4 0C | F5 03 | F6 0B | F7 83 | F8 0A | F9 01 | F10 09 | F11 78 | F12 07

~` 0E | 1! 16 | 2@ 1E | 3# 26 | 4$ 25 | 5% 2E | 6^ 36 | 7& 3D | 8 * 3E | 9 (46 | 0) 45 | -_ 4E | + = 55 | |\ 5D | ← 66

TAB 0D | Q 15 | W 1D | E 24 | R 2D | T 2C | Y 35 | U 3C | I 43 | O 44 | P 4D | [{ 54 | }] 5B

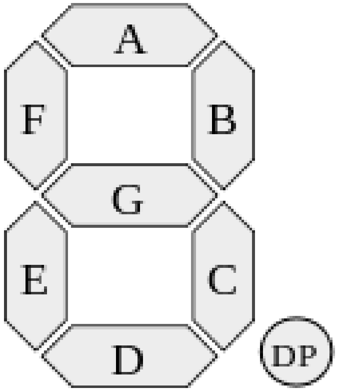
Caps 58 | A 1C | S 1B | D 23 | F 2B | G 34 | H 33 | J 3B | K 42 | L 4B | : ; 4C | " ' 52 | ↵ 5A

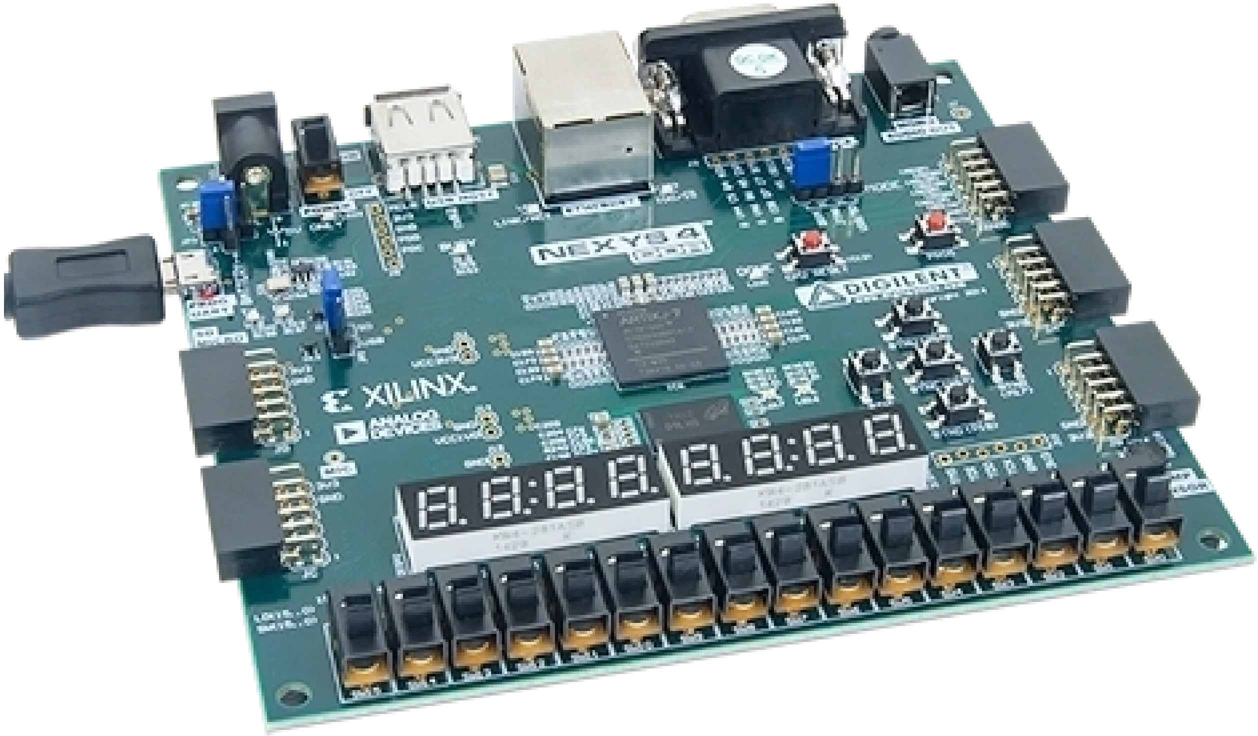
Shift 12 | Z 1A | X 22 | C 21 | V 2A | B 32 | N 31 | M 3A | < , 41 | > . 49 | ? / 4A | Shift 59

Ctrl 14 | Alt 11 | SPACE 29 | Alt E0 11 | Ctrl E0 14

FPGA

- **Pressed key shows on FPGA -- 7 SEGEMENT DISPLAY**





Demonstration of you design

- Check the reset condition, all display should be switched off at reset.
- Pressing "123456" one by one, the display should look the following:

Seven Segement Display

DISPLAY OFF				press (sw0) reset
			1	press 1
		1	2	press 2
	1	2	3	press 3
1	2	3	4	press 4
2	3	4	5	press 5
3	4	5	6	press 6

Important Concepts

- The frequency difference between FPGA and keyboard
  (keyboard freq. typically is a few KHz (~10-30) and FPGA in the MHz range)

Synchronization

between two domains is the key to avoid meta-stability

- Keyboard operates by sampling the data line on the falling edge of its clock

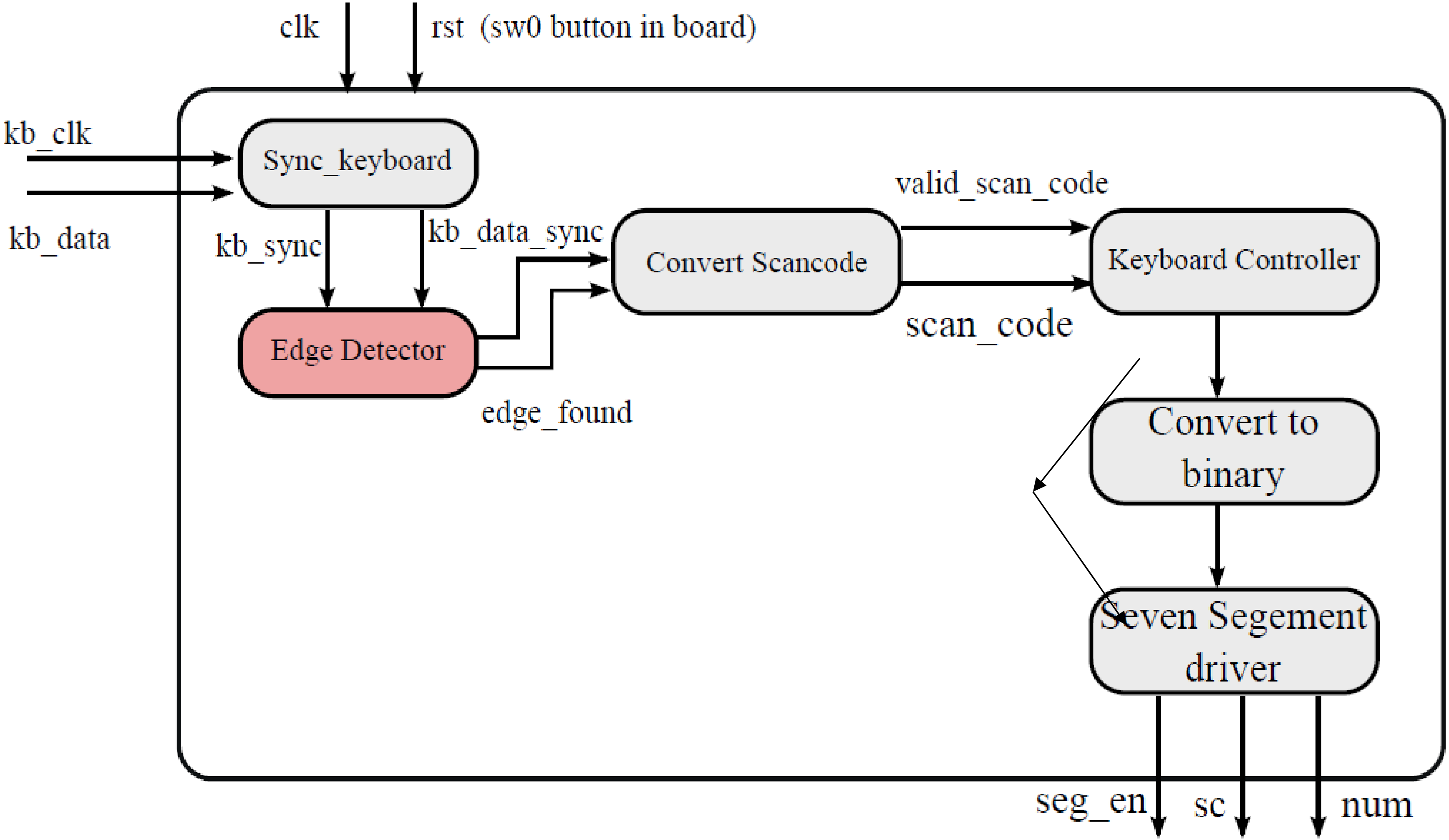
Detecting falling edge

of keyboard clock is the key not to miss any data from data line

Main Processes

- Synchronizing Keyboard with FPGA
- Detection of falling edge of keyboard Clock
- Storing of relevant Scan Code from keyboard
- Control and activate the required seven segment
- Shift previous key left and display the currently pressed key

System blocks



Tips and Hints

- Build your own test bench for each component & TEST It!

**The sooner you understand any problem in your design,
The easier you can solve it and go along the flow**

- Comment your code wherever possible!

- Document the your work process!

- Make sure you keep the input/output ports of each component the same as they are provided.

Though you can create your own internal signals in the components.

Tips and Hints

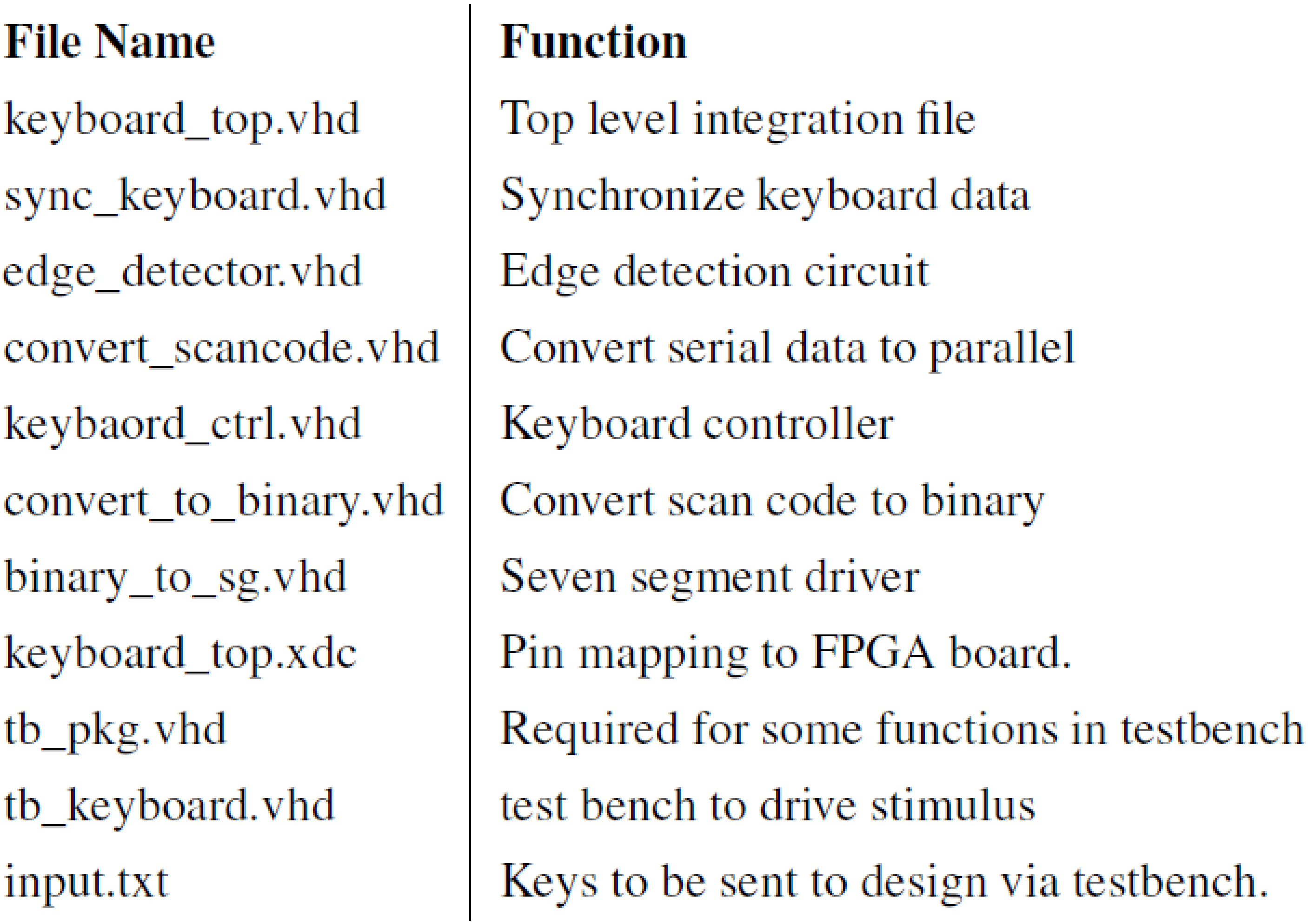
- **Read the required parts from reference manual of FPGA board. You need to understand e.g., how to refresh the 7segments to show your digits**

- **Almost all necessary solutions to your problems can be found in the course book. Also, searching through internet may help you for better understanding**

- **And Finally:**

  No Latch!

VHDL File Lists



File Name	Function
keyboard_top.vhd	Top level integration file
sync_keyboard.vhd	Synchronize keyboard data
edge_detector.vhd	Edge detection circuit
convert_scancode.vhd	Convert serial data to parallel
keybaord_ctrl.vhd	Keyboard controller
convert_to_binary.vhd	Convert scan code to binary
binary_to_sg.vhd	Seven segment driver
keyboard_top.xdc	Pin mapping to FPGA board.
tb_pkg.vhd	Required for some functions in testbench
tb_keyboard.vhd	test bench to drive stimulus
input.txt	Keys to be sent to design via testbench.