



# Keyboard

黃元豪

Yuan-Hao Huang

國立清華大學電機工程學系

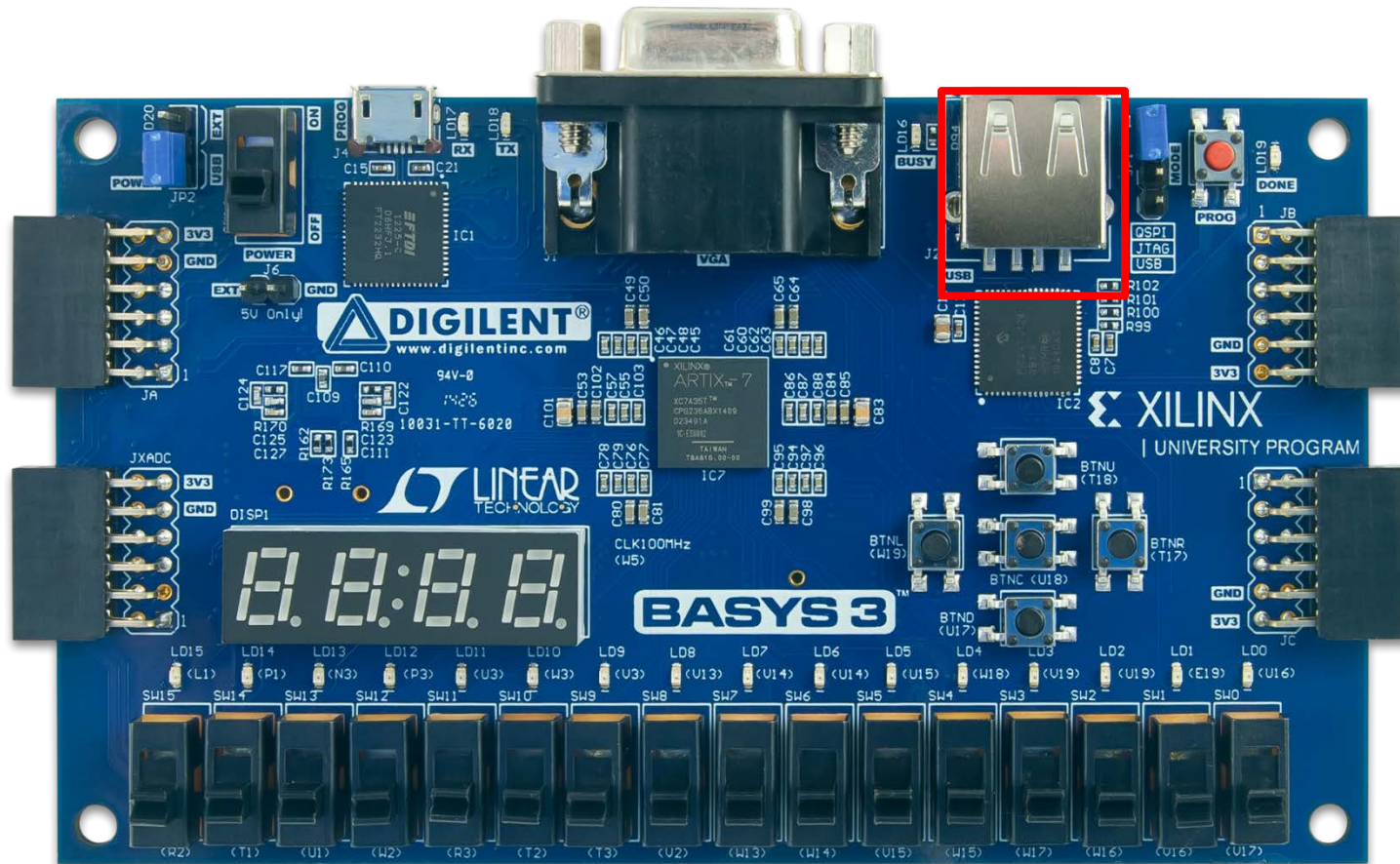
Department of Electrical Engineering

National Tsing-Hua University



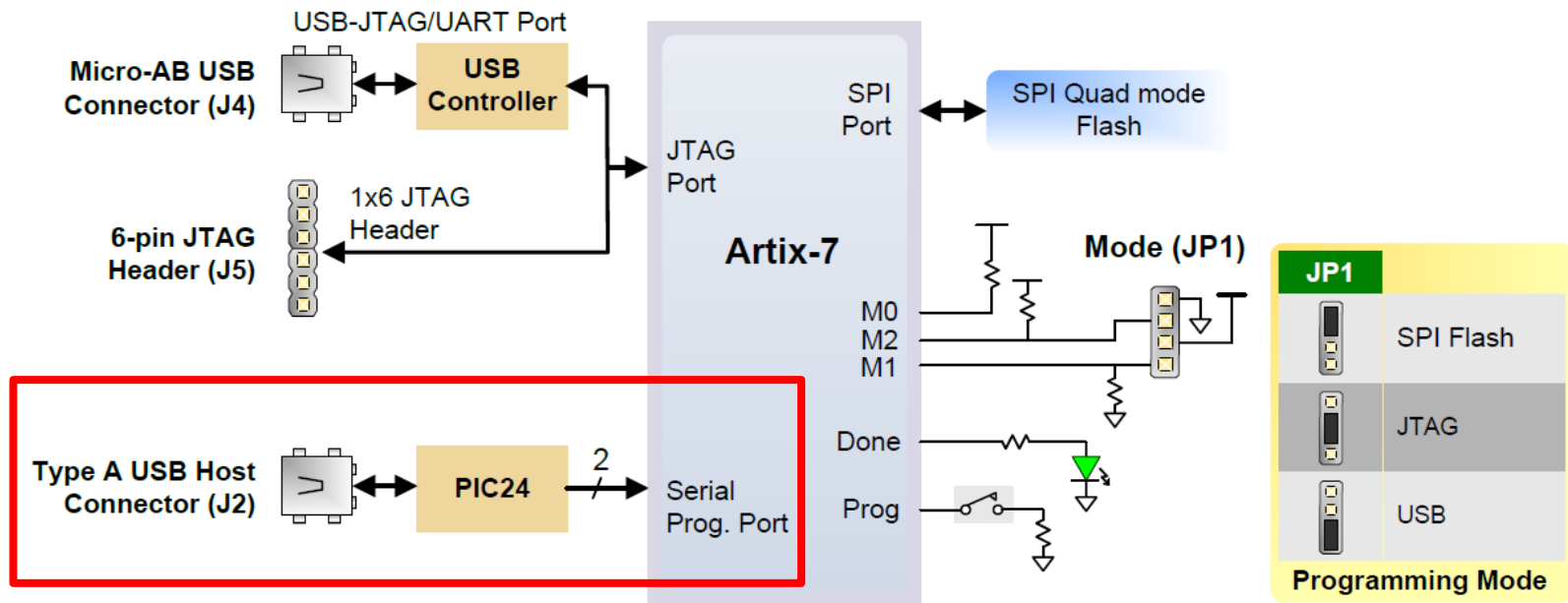
# USB HID Host (1/3)

HID : Human Interface Device

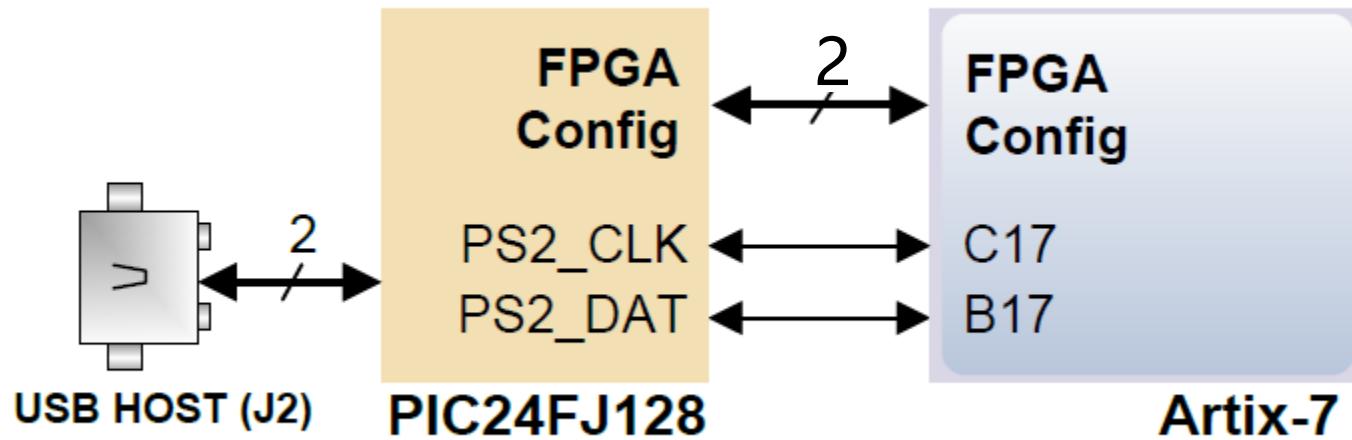




# USB HID Host (2/3)



# USB HID Host (3/3)

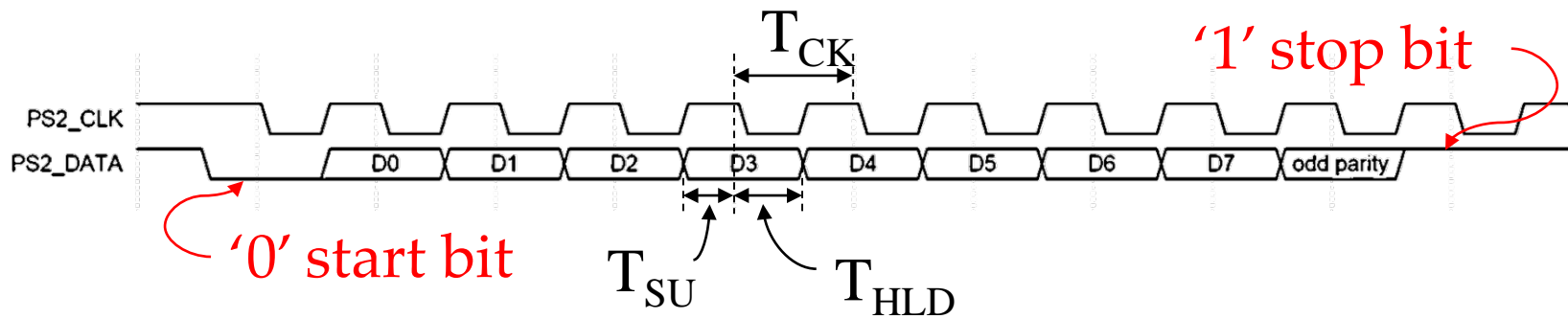


# Microchip PIC24FJ128



- Configuration mode
  - Download a bit-stream to the FPGA.
- Application mode
  - In Basys 3, this is called USB HID Host mode.
  - Only a single mouse or a single keyboard can be used.
  - PS2\_CLK and PS2\_DATA are used to implement **a standard PS/2 interface.**

# HID Controller



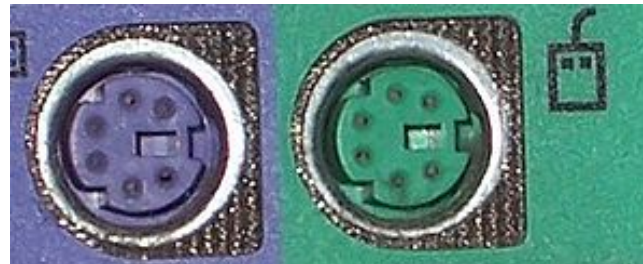
Symbol	Parameter	Min	Max
$T_{CK}$	Clock time	30 us	50 us
$T_{SU}$	Data-to-clock setup time	5 us	25 us
$T_{HLD}$	Clock-to-data hold time	5 us	25us



# Initialization

- Initially, Basys3 identifies the devices through PS2\_CLK and PS2\_DATA.
- When Basys3 is idled (unconnected), Basys3 reads 0xFA using a Read ID command.
- When a keyboard or mouse is connected to the Basys3, a “self-test passed” command (0xAA) is sent to the Basys3.
  - 0xFA → 0xAA
- Scancode of keyboard
  - Each key is assigned a code
  - If the key is held down, the scan code will be sent repeatedly about once every 100ms.
  - When a key is released, an F0 key-up code is sent, followed by the scan code of the released key.
  - Some keys (right Ctrl, right Alt, ...) , called extended keys, send an E0 ahead of the scan code.

# PS/2 Port



Example PC compatible (IBM PS/2) scancodes

key	set 1 (IBM PC XT)		set 2 (IBM PC AT)		set 3 (IBM 3270 PC)	
	press	release	press	release	press	release
A (normal letter)	1E	9E	1C	F0 1C	1C	F0 1C
Return / Enter (main keyboard)	1C	9C	5A	F0 5A	5A	F0 5A
Enter (numeric keypad)	E0 1C	E0 9C	E0 5A	E0 F0 5A	79	F0 79
Left Windows key	E0 5B	E0 DB	E0 1F	E0 F0 1F	8B	F0 8B
Right Windows key	E0 5C	E0 DC	E0 27	E0 F0 27	8C	F0 8C

from Wiki

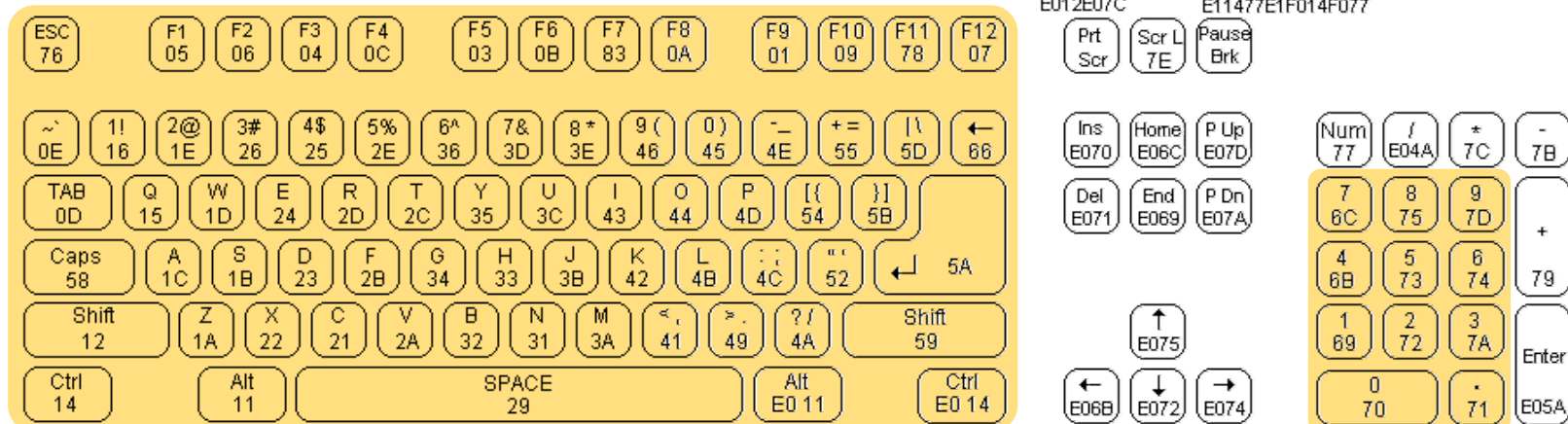




# PS/2 Scancode

Extend Code	Break Code	Make code
E0	F0	XX

(means “release”)

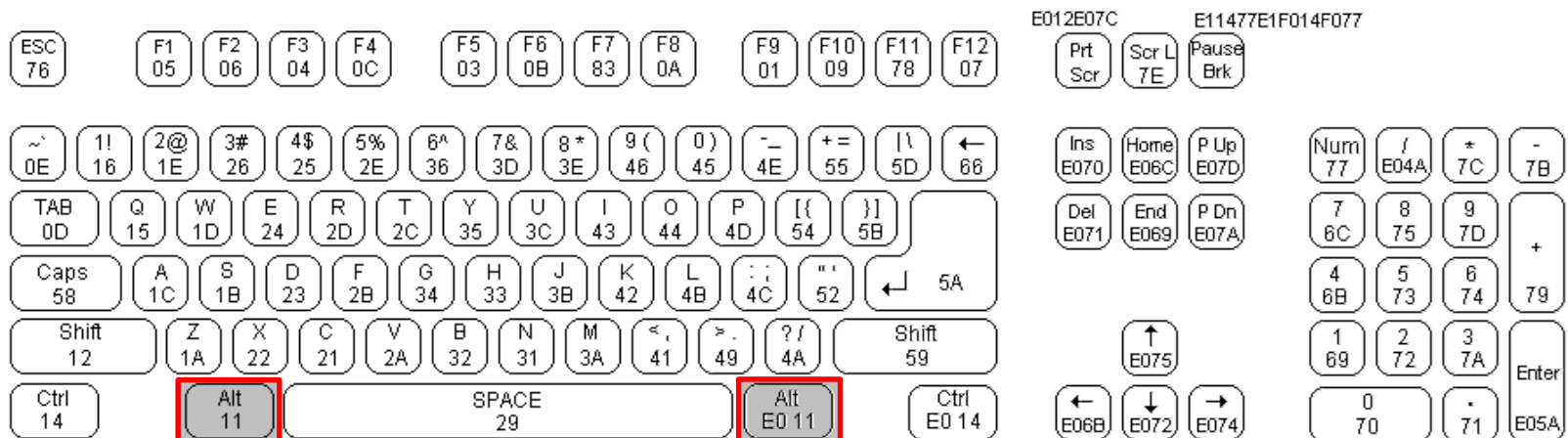


We only use the yellow parts of the keyboard.



# PS/2 Scancode (Example)

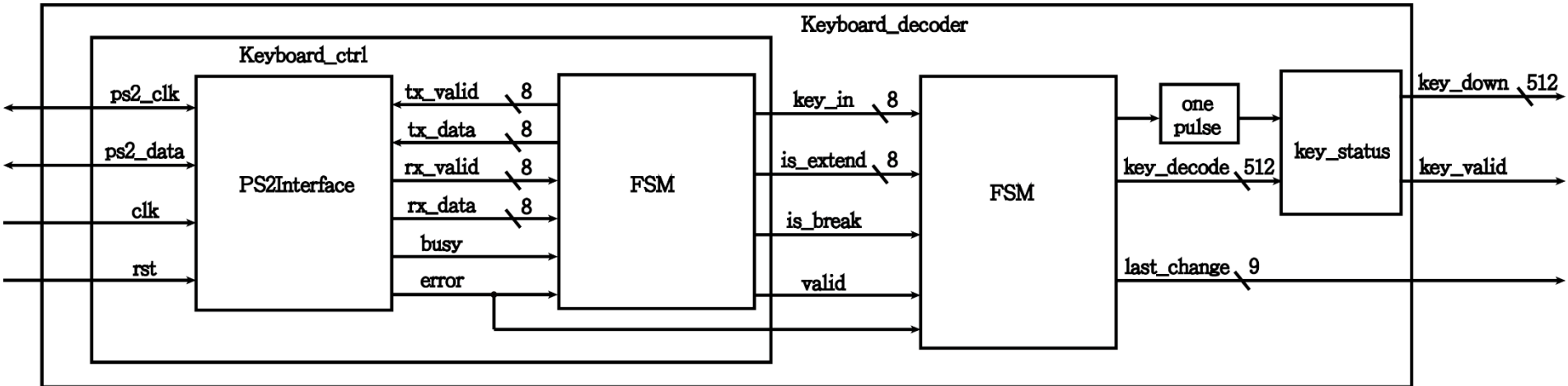
<b>L Alt press</b>			11
<b>L Alt release</b>		F0	11
<b>R Alt press</b>	E0		11
<b>R Alt release</b>	E0	F0	11



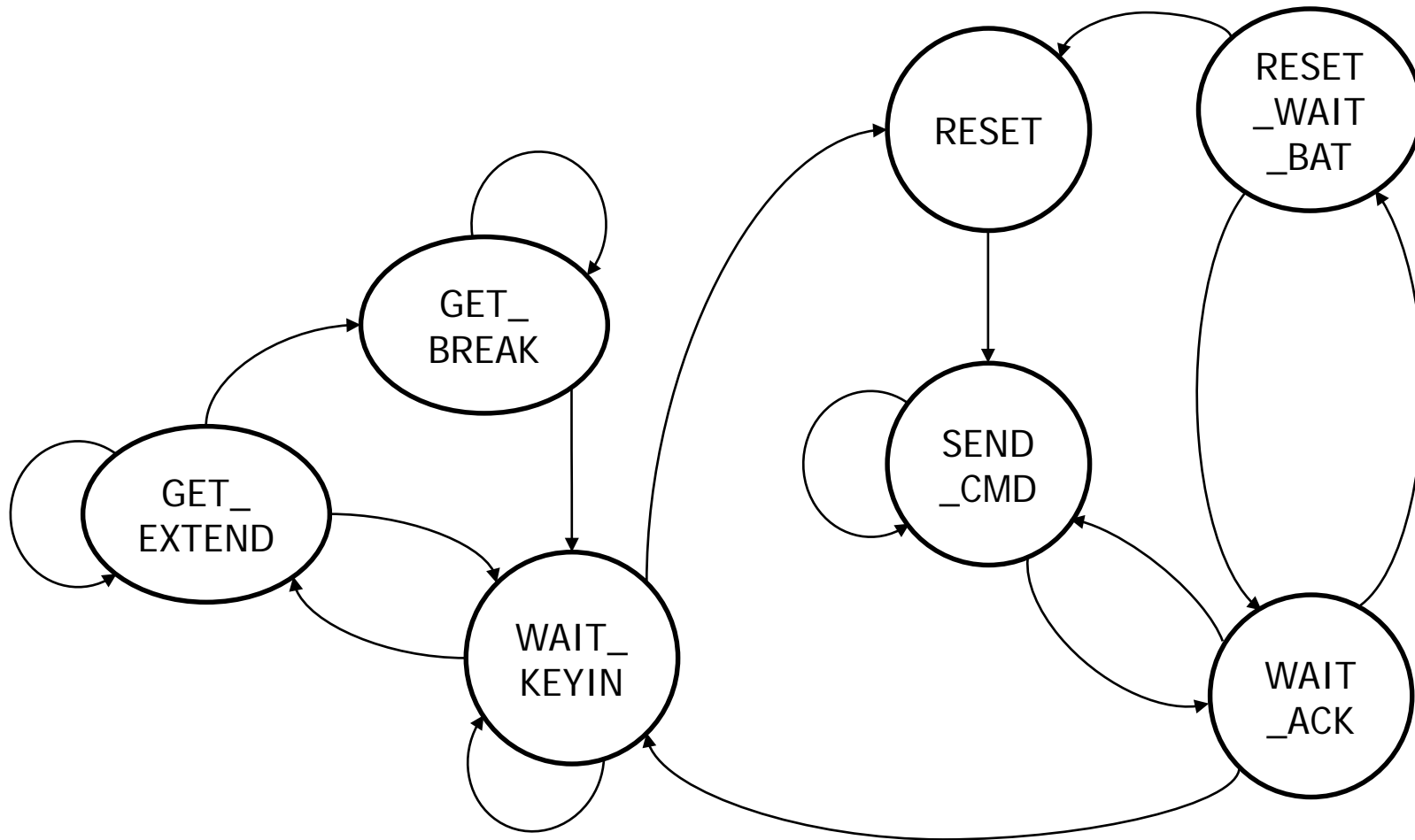


# Verilog Module: KeyboardCtrl(1/2)

- In Keyboard-Controller
  - Ps2Interface.v
  - KeyboardCtrl.v
- KeyboardCtrl.v
  - **Input** : PS2\_CLK, PS2\_DATA, rst, clk
  - **Output** : key\_in, is\_extend, is\_break, valid, err



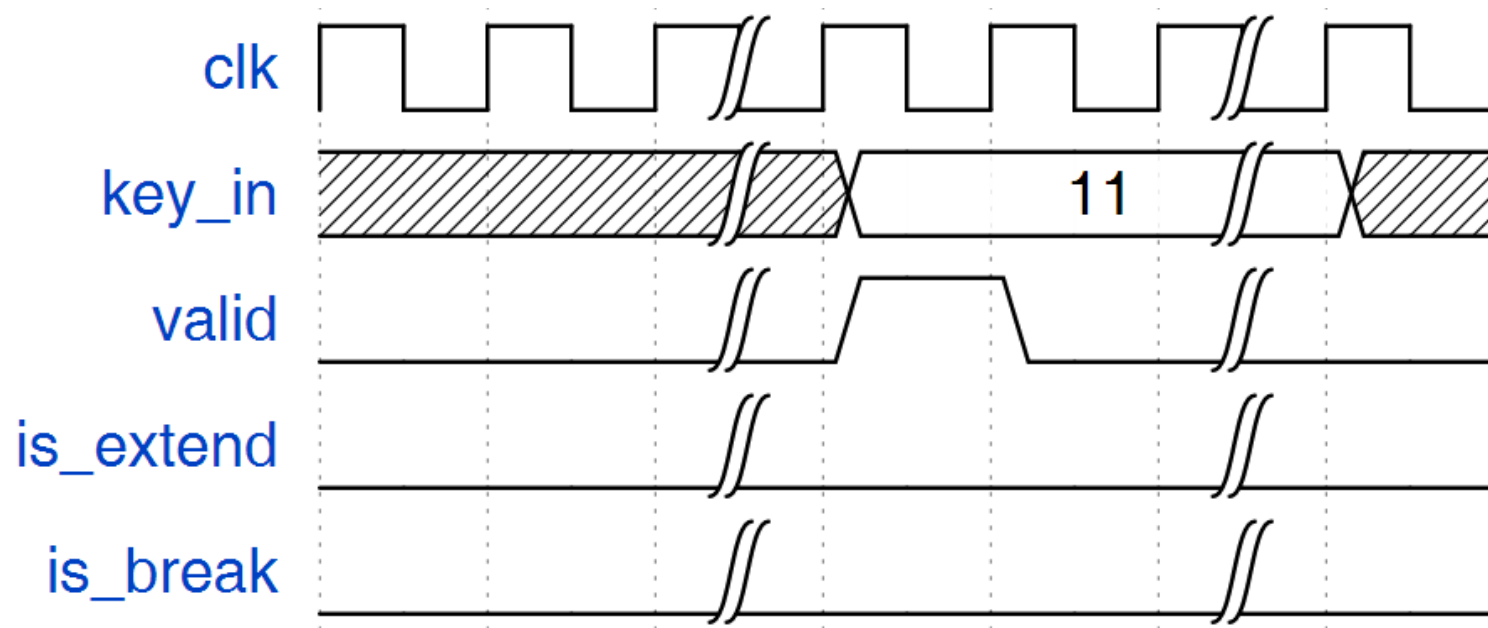
# Verilog Module: KeyboardCtrl(1/2)



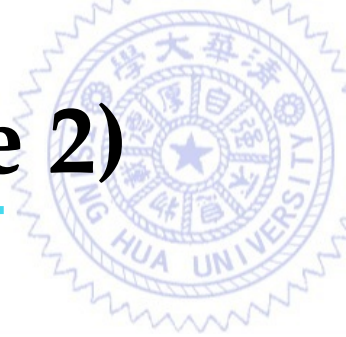
# KeyboardCtrl (Output Example 1)



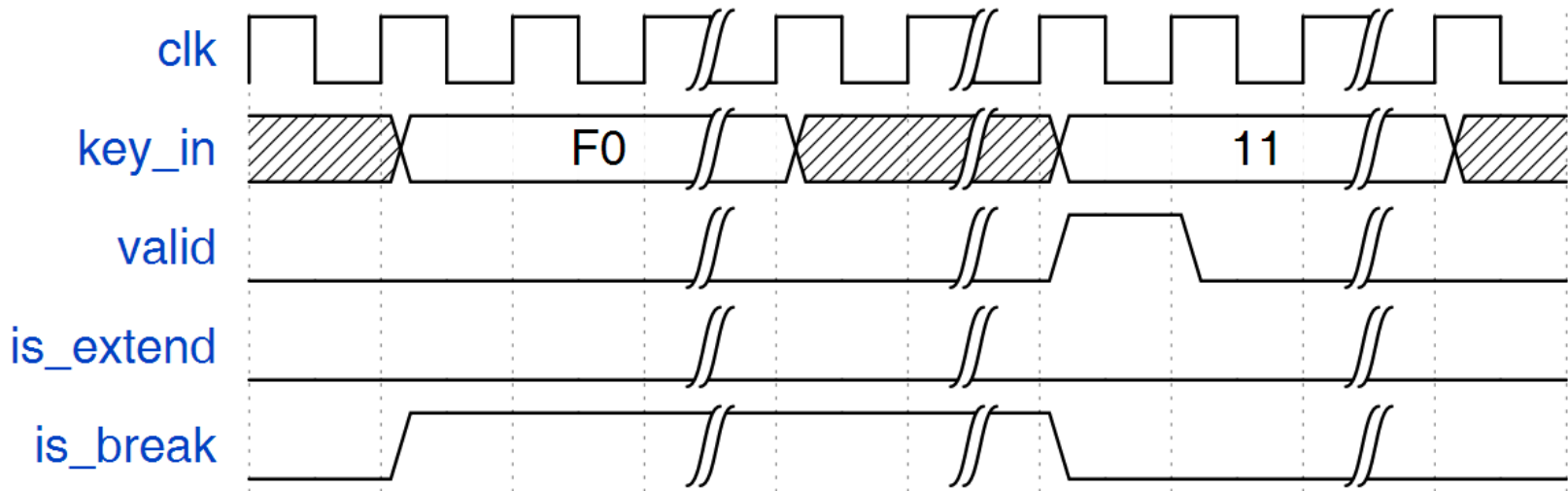
## L Alt press



# KeyboardCtrl (Output Example 2)



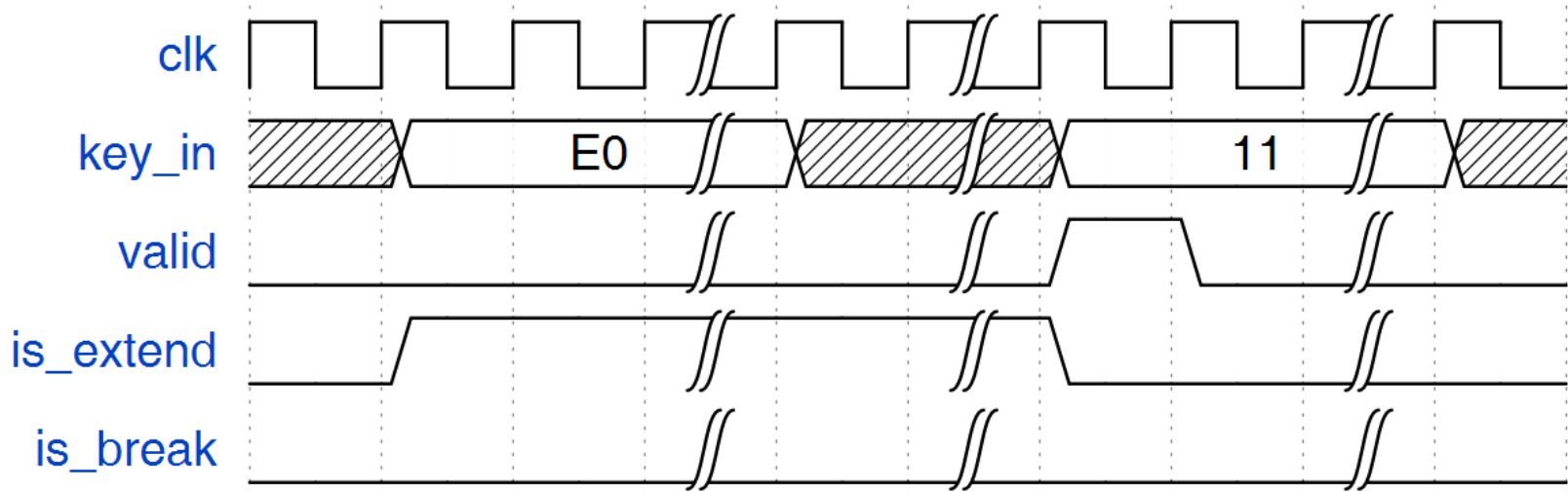
## L Alt release



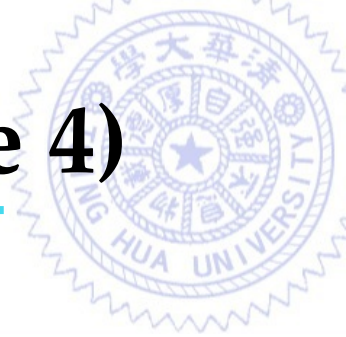
# KeyboardCtrl (Output Example 3)



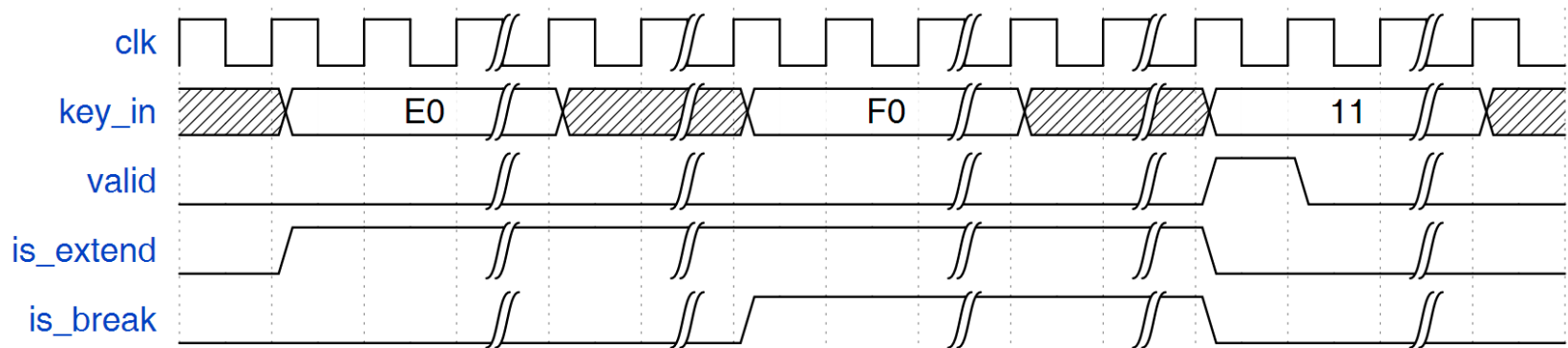
## R Alt press



# KeyboardCtrl (Output Example 4)



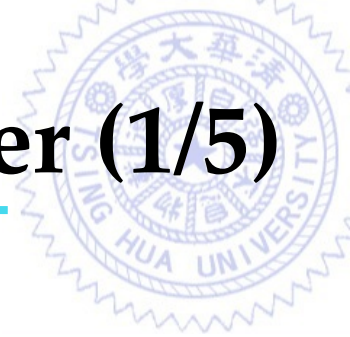
## R Alt release





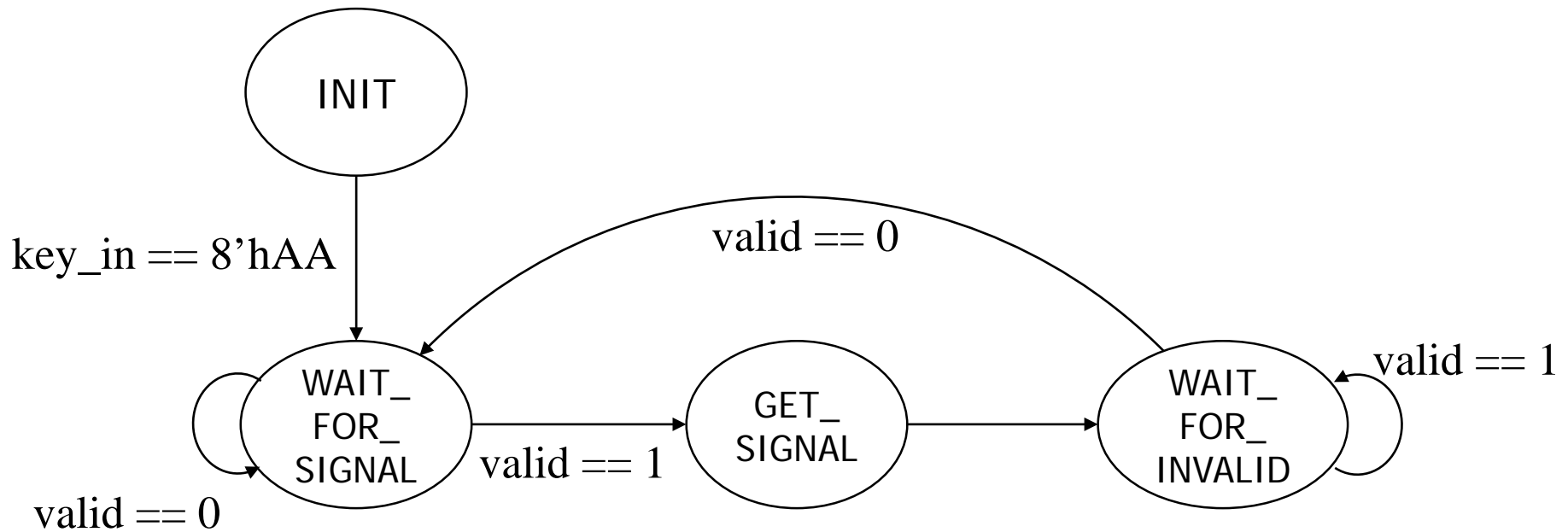
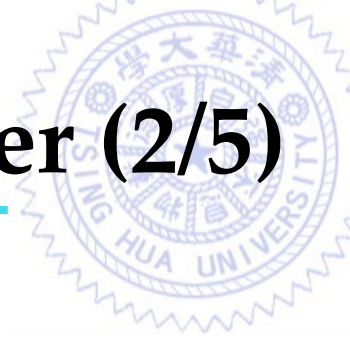
# Verilog Module: KeyboardDecoder (1/5)

---



- In Keyboard Sample Code
  - KeyboardDecoder.v
- I/O for KeyboardDecoder
  - Input : PS2\_CLK, PS2\_DATA, rst, clk
  - Output : key\_down, last\_change, Key\_valid

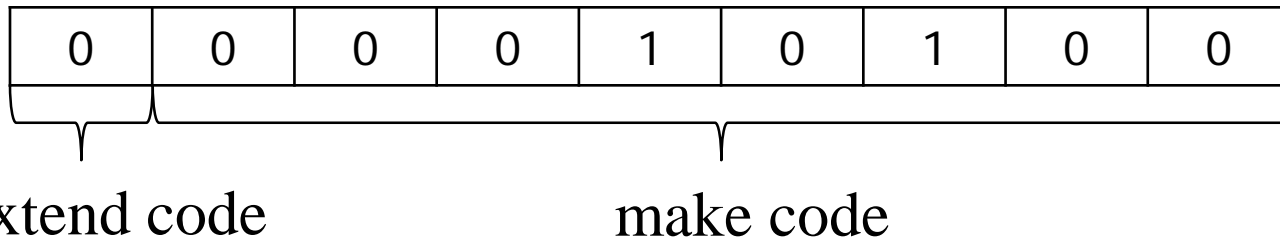
# Verilog Module: KeyboardDecoder (2/5)



# Verilog Module: KeyboardDecoder (3/5)



- last\_change: 9 bits
  - represent the key which has been pressed or released.

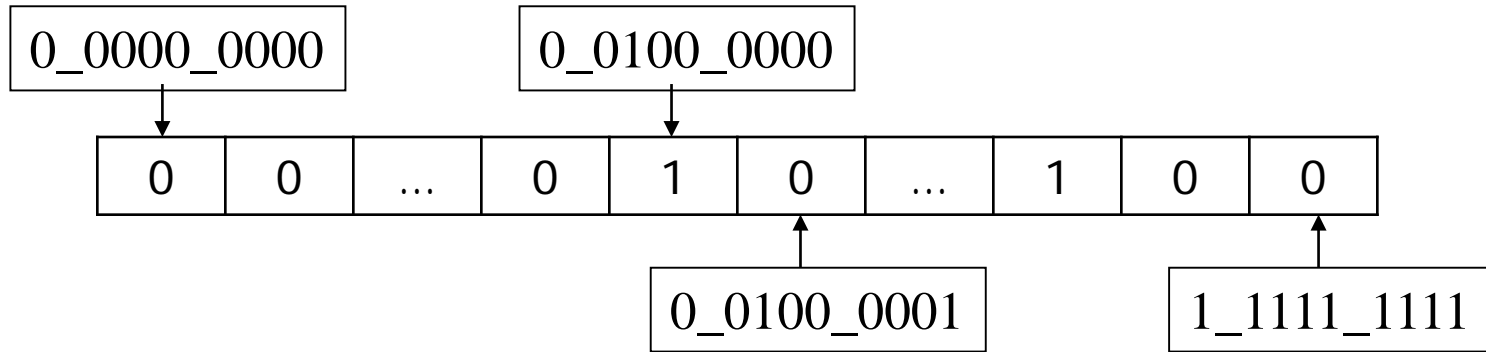
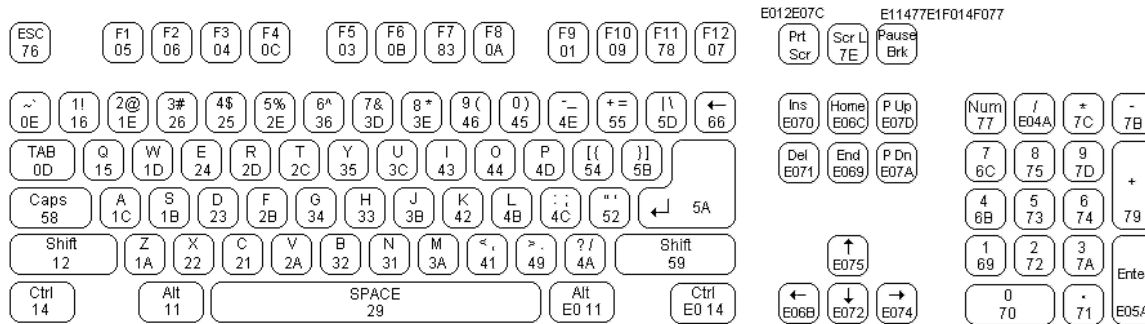


- key\_valid: 1 bit
  - should be active for one clock period (100MHz) when any key is pressed or released.



# Verilog Module: KeyboardDecoder (4/5)

- key\_down [511:0] are status bits. Each bit indicates pressed (1) or released (0) of each button of the keyboard.



- the key indexed by "0\_0100\_0000" is pressed.
- the key indexed by "0\_0100\_0001" is released.

# Verilog Module: KeyboardDecoder (5/5)



- `key_down [511:0]`
- `key_down <= key_down | key_decode;`

	0	1	1	0	1
or	0	0	0	1	0
<hr/>					
	0	1	1	1	1

- `key_down <= key_down & (~key_decode);`

	0	1	1	0	1
and	1	1	0	1	1
<hr/>					
	0	1	0	0	1



# How to Use IP (1/3)

The screenshot illustrates the steps to add a new IP repository in the IP Integrator software. The interface is divided into several panels:

- Flow Navigator:** Shows the project workflow from Project Manager to Implementation to Bitstream.
- Project Manager:** Contains icons for Project Settings (1), Add Sources, Language Templates, and IP Catalog.
- Project Settings:** The 'IP' tab is active. It shows a list of IP Repositories. A red circle (3) highlights the '+' button to add a new repository. A red circle (4) highlights the 'Your IP' directory in the list.
- IP Repositories Dialog:** A file explorer window showing the directory structure. A red circle (5) highlights the 'Select' button at the bottom right.
- Bottom Panel:** A red circle (6) highlights the 'OK' button.

# How to Use IP (2/3)



The screenshot illustrates the process of using an IP block in Vivado. It is divided into several key areas:

- Project Manager:** The 'IP Catalog' option is highlighted with a red circle and the number 7.
- IP Catalog:** The 'KeyboardCtrl\_v1\_0' IP block is selected in the 'User Repository' and highlighted with a red circle and the number 8.
- Customize IP:** The 'KeyboardCtrl\_v1\_0 (1.0)' configuration window is shown, displaying the component name and system clock frequency. A red circle with the number 9 is placed near the 'OK' button at the bottom right.
- Generate Output Products:** A dialog box shows the output products to be generated, including the instantiation template, synthesized checkpoint, and behavioral simulation. The 'Generate' button is highlighted with a red circle and the number 10.

# How to Use IP (3/3)

