PS/2 Keyboard to ASCII Converter (VHDL)

Scott_1767 DigiKey Employee Mar '21

Logic Home

Code Downloads

PS/2 Keyboard to ASCII Converter Logic (top level VHDL for the design): ps2_keyboard_to_ascii.vhd (16.9 KB)

PS/2 Keyboard Interface Logic (must be included in the project): ps2_keyboard.vhd (5.1 KB)

Debounce Logic (must be included in the project): debounce.vhd (2.4 KB)

Features

- VHDL source code of a PS/2 keyboard to ASCII converter
- Outputs the ASCII codes that correspond to key presses on a PS/2 keyboard
- Ignores keys that have no ASCII equivalent
- Sets a flag output when a new ASCII code is available
- Validates parity, start, and stop bits of PS/2 transactions
- Configurable system clock frequency
- Synchronizes between PS/2 and system clock domains
- Debounces incoming PS/2 signals

Introduction

This details a PS/2 keyboard to ASCII converter component for use in CPLDs and FPGAs, written in VHDL. The component receives data transactions from a PS/2 keyboard and provides the corresponding ASCII codes to user logic over a parallel interface. It provides codes only for key presses (not releases) and ignores keys that do not have ASCII equivalents. It was designed using Quartus II, version 12.1. Figure 1 illustrates a typical example of the PS/2 keyboard to ASCII converter integrated into a system.

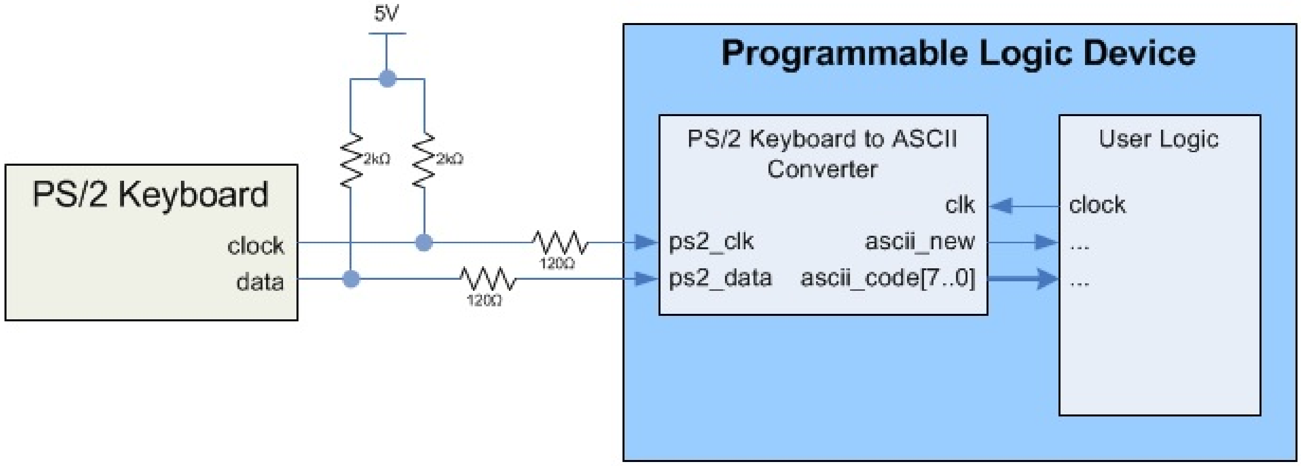


Figure 1. Example Implementation

Background

PS/2 Keyboards

PS/2 (IBM Personal System/2) is an interface for keyboards and mice to PC compatible computer systems via a 6-pin Mini-DIN connector. The computing system must provide the keyboard or mouse with 5V source and ground connections. Communication occurs over a 2-wire serial interface, consisting of a clock line and a data line. Both lines require pull-up resistors (2kΩ shown in Figure 1). The 120Ω series resistors in Figure 1 are required to interface the 3.3V FPGA I/O to the 5V signals.

Once powered, the keyboard goes through a self- initialization sequence. Upon completion, it is ready to communicate keyboard events over the PS/2 interface.

Figure 2 illustrates the transaction format. Both clock and data signals are logic level high when inactive (note pull-up resistors). The keyboard provides both the clock and data. The clock has a frequency between 10 kHz and 16.7 kHz (i.e. a 60-100us period). The data begins with a start bit (logic low), followed by one byte of data, a parity bit, and finally a stop bit (logic high). The data is sent LSB first. Each bit should be read on the falling edge of the clock signal. Once complete, both the clock and data signals return to logic level high.

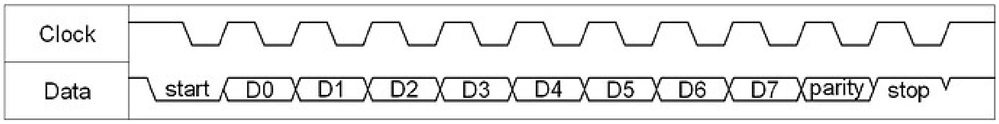


Figure 2. PS/2 Keyboard Transmission Timing Diagram

The data byte represents part of a keyboard scan code: either a make code (key press) or a break code (key release). Three different sets of scan codes exist, however the vast majority of keyboards use Scan Code Set 2, which is provided in the Appendix on **this page** . A make code usually consists of either one or two bytes. If a make code uses two bytes, the first byte is x“E0”. [redacted] is typically the same as its make code, except that break codes include an additional x“F0” byte as the 2nd to last byte. (The PAUSE and PRNT SCRN keys are exceptions to the above.)

While it is possible to send data to a keyboard (to change its settings, etc.), this is generally unnecessary and is not included in this VHDL component.

ASCII

ASCII (American Standard Code for Information Interchange) is a prominent standard for encoding characters and control codes into 7-bit binary integers. ASCII defines 128 individual codes (95 characters and 33 control codes). These codes can all be translated from a keyboard, either as direct key presses or as combinations of key presses.

PS/2 to ASCII Conversion

This PS/2 keyboard to ASCII converter handles incoming transactions from the PS/2 keyboard and determines which keys are being pressed at a given time. It then outputs the ASCII encoding that corresponds to these key presses. The appendix lists the ASCII Character Set. All character keys have their respective ASCII codes outputted by the PS/2 to ASCII converter as listed, when those keys are pressed in a standard fashion (with respect to shift, caps lock, etc.). If a control key is held down, the converter outputs the corresponding control code for each key instead (irrespective of shift or caps lock). The converter also outputs control codes for the additional following keys: backspace (BS), tab (HT), enter (CR), escape (ESC), and delete (DEL).

Theory of Operation

The source code for the PS/2 keyboard to ASCII converter consists of the following three files:

- ps2_keyboard_to_ascii.vhd
- ps2_keyboard.vhd
- debounce.vhd

The ps2_keyboard_to_ascii.vhd file is the top level VDHL. It instantiates the PS/2 keyboard interface component (ps2_keyboard.vhd). This internal component handles the transactions with the keyboard. It synchronizes the interface and system clocks domains, debounces the input signals, performs error checking, and provides the codes received from the keyboard. Detailed documentation for the PS/2 keyboard interface is available **here** . (The PS/2 keyboard interface component in turn instantiates the debounce component, debounce.vhd. Documentation for the debounce component is available **here** .)

The PS/2 codes provided by the PS/2 keyboard interface component control the converter’s state machine. Figure 3 depicts the operation of the state machine. Upon start-up, the component immediately enters the *ready* state. It waits in this state until it receives a new PS/2 code. The *new_code* state builds the PS/2 make or break codes. If the new code received is the last byte in the make/break code, the state machine proceeds to the *translate* state, otherwise it returns to the [redacted]ext byte. Once in the *translate* state, the converter determines which key es it into ASCII. If the code is a break code, then no action is needed, so the converter ignores the code and returns to the *ready* state. However, if a make code was received, the converter proceeds to the *output* state, where it outputs the resultant ASCII code on the *ascii_code* bus and sets the *ascii_new* output flag to indicate that the new code is available. The converter then returns to the *ready* state to await the next communication from the PS/2 keyboard interface component.

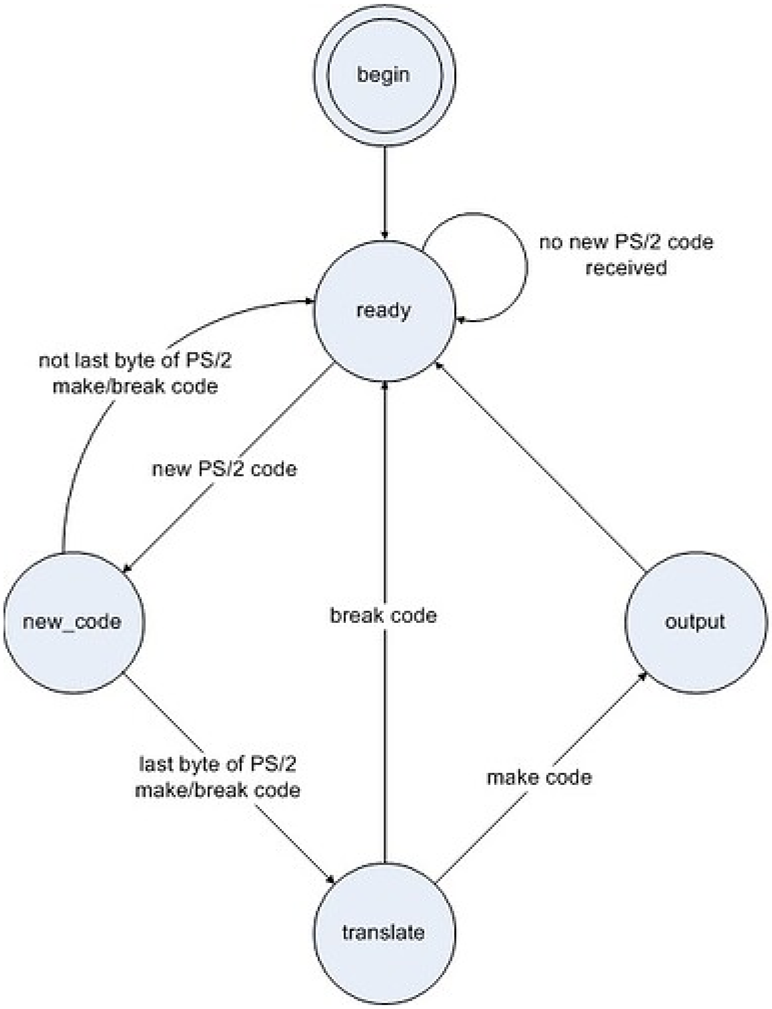
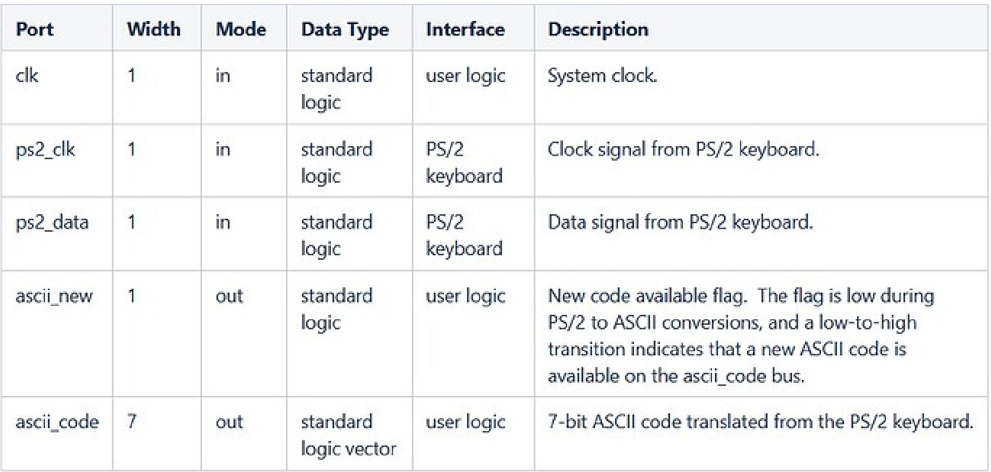

begin
no new PS/2 code received
ready
not last byte of PS/2 make/break code
new PS/2 code
new_code
break code
output
last byte of PS/2 make/break code
make code
translate

Figure 3. State Diagram

Port Descriptions

Table 1 describes the PS/2 keyboard to ASCII converter's ports.

Table 1. Port Descriptions



Port	Width	Mode	Data Type	Interface	Description
clk	1	in	standard logic	user logic	System clock.
ps2_clk	1	in	standard logic	PS/2 keyboard	Clock signal from PS/2 keyboard.
ps2_data	1	in	standard logic	PS/2 keyboard	Data signal from PS/2 keyboard.
ascii_new	1	out	standard logic	user logic	New code available flag. The flag is low during PS/2 to ASCII conversions, and a low-to-high transition indicates that a new ASCII code is available on the ascii_code bus.
ascii_code	7	out	standard logic vector	user logic	7-bit ASCII code translated from the PS/2 keyboard.

Setting the Timing Parameters

The system clock speed affects the debounce time and the idle counter timing of the PS/2 keyboard interface component within the converter. The two GENERIC parameters declared in the ENTITY, *clk_freq* and *ps2_debounce_counter_size* must be set appropriately for the component to operate correctly. The *clk_freq* parameter must be set to the system clock *clk* frequency in Hz. The default setting in the provided code is 50 MHz (the frequency at which the component was simulated and tested). The *ps2_debounce_counter_size* parameter must be set such that 2^*ps2_debounce_counter_size* / *clk_freq* = 5us, as described in the documentation for the debounce component **here** . For a 50 MHz system clock, *ps2_debounce_counter_size* = 8.

Example Transaction

Figure 4 shows the timing diagram of an example transaction. Once the internal PS/2 keyboard interface finishes receiving the PS/2 transaction, the *ascii _new* flag deasserts to indicate that a new PS/2 to ASCII conversion is in progress. When the transaction completes, the *ascii_new* flag asserts to indicate that a new ASCII code has been received and is available on the *ascii_code* bus. In this case, the PS/2 code received is x“1C” (the make code for the “A” key), and the resulting ASCII code is x“61” (the ASCII code for the “a” character).

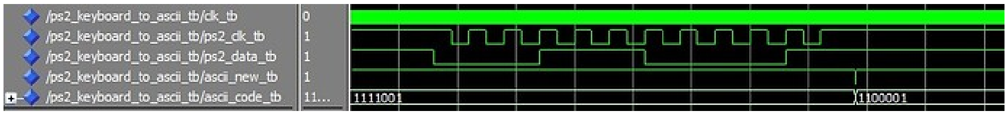

/ps2_keyboard_to_ascii_tb/clk_tb 0
/ps2_keyboard_to_ascii_tb/ps2_clk_tb 1
/ps2_keyboard_to_ascii_tb/ps2_data_tb 1
/ps2_keyboard_to_ascii_tb/ascii_new_tb 1
/ps2_keyboard_to_ascii_tb/ascii_code_tb 11...
1111001
1100001

Figure 4. Example Transaction

Conclusion

This PS/2 keyboard to ASCII converter is a programmable logic component that receives transactions from PS/2 keyboards and outputs their ASCII equivalents. It synchronizes the PS/2 and system clock domains, debounces the input signals, performs error checking, translates the PS/2 codes into ASCII codes, and notifies the user logic when new ASCII codes are available on i

Appendix: ASCII Character Set

Table A1. ASCII Character Set

Control Codes

Binary	Dec	Hex	Control Code	Key	Control Code Name
000 0000	0	0	NUL	^@	Null Character
000 0001	1	1	SOH	^A	Start of Header
000 0010	2	2	STX	^B	Start of Text
000 0011	3	3	ETX	^C	End of Text
000 0100	4	4	EOT	^D	End of Transmission
000 0101	5	5	ENQ	^E	Enquiry
000 0110	6	6	ACK	^F	Acknowledgment
000 0111	7	7	BEL	^G	Bell
000 1000	8	8	BS	^H	Backspace
000 1001	9	9	HT	^I	Horizontal Tab
000 1010	10	0A	LF	^J	Line feed
000 1011	11	0B	VT	^K	Vertical Tab
000 1100	12	0C	FF	^L	Form Feed
000 1101	13	0D	CR	^M	Carriage Return
000 1110	14	0E	SO	^N	Shift Out
000 1111	15	0F	SI	^O	Shift In
001 0000	16	10	DLE	^P	Data Link Escape
001 0001	17	11	DC1	^Q	Device Control 1 (oft. XON)
001 0010	18	12	DC2	^R	Device Control 2
001 0011	19	13	DC3	^S	Device Control 3 (oft. XOFF)
001 0100	20	14	DC4	^T	Device Control 4
001 0101	21	15	NAK	^U	Negative Acknowledgement
001 0110	22	16	SYN	^V	Synchronous Idle
001 0111	23	17	ETB	^W	End of Transmission Block
001 1000	24	18	CAN	^X	Cancel
001 1001	25	19	EM	^Y	End of Medium
001 1010	26	1A	SUB	^Z	Substitute
001 1011	27	1B	ESC	^[	Escape
001 1100	28	1C	FS	^\	File Separator
001 1101	29	1D	GS	^]	Group Separator

Control Codes

001 1110	30	1E	RS	^^	Record Separator
001 1111	31	1F	US	^_	Unit Separator
111 1111	127	7F	DEL	^?	Delete

Character Codes

Binary	Dec	Hex	Character
010 0000	32	20	
010 0001	33	21	!
010 0010	34	22	"
010 0011	35	23	#
010 0100	36	24	$
010 0101	37	25	%
010 0110	38	26	&
010 0111	39	27	’
010 1000	40	28	(
010 1001	41	29	)
010 1010	42	2A	*
010 1011	43	2B	+
010 1100	44	2C	,
010 1101	45	2D	-
010 1110	46	2E	.
010 1111	47	2F	/
011 0000	48	30	0
011 0001	49	31	1
011 0010	50	32	2
011 0011	51	33	3
011 0100	52	34	4
011 0101	53	35	5
011 0110	54	36	6
011 0111	55	37	7
011 1000	56	38	8
011 1001	57	39	9
[redacted]	[redacted]	3A	:
[redacted]	[redacted]	3B	;

Character Codes

011 1100	60	3C	<
011 1101	61	3D	=
011 1110	62	3E	>
011 1111	63	3F	?
100 0000	64	40	@
100 0001	65	41	A
100 0010	66	42	B
100 0011	67	43	C
100 0100	68	44	D
100 0101	69	45	E
100 0110	70	46	F
100 0111	71	47	G
100 1000	72	48	H
100 1001	73	49	I
100 1010	74	4A	J
100 1011	75	4B	K
100 1100	76	4C	L
100 1101	77	4D	M
100 1110	78	4E	N
100 1111	79	4F	O
101 0000	80	50	P
101 0001	81	51	Q
101 0010	82	52	R
101 0011	83	53	S
101 0100	84	54	T
101 0101	85	55	U
101 0110	86	56	V
101 0111	87	57	W
101 1000	88	58	X
101 1001	89	59	Y
101 1010	90	5A	Z
101 1011	91	5B	[
		5C	\

Character Codes

101 1101	93	5D	]
101 1110	94	5E	^
101 1111	95	5F	_
110 0000	96	60	`
110 0001	97	61	a
110 0010	98	62	b
110 0011	99	63	c
110 0100	100	64	d
110 0101	101	65	e
110 0110	102	66	f
110 0111	103	67	g
110 1000	104	68	h
110 1001	105	69	i
110 1010	106	6A	j
110 1011	107	6B	k
110 1100	108	6C	l
110 1101	109	6D	m
110 1110	110	6E	n
110 1111	111	6F	o
111 0000	112	70	p
111 0001	113	71	q
111 0010	114	72	r
111 0011	115	73	s
111 0100	116	74	t
111 0101	117	75	u
111 0110	118	76	v
111 0111	119	77	w
111 1000	120	78	x
111 1001	121	79	y
111 1010	122	7A	z
111 1011	123	7B	{
111 1100	124	7C	
[redacted]	[redacted]	7D	}

Character Codes			
111 1110	126	7E	~

Related Topics

PS/2 Host Transceiver (VHDL)
PS/2 Keyboard Interface (VHDL)
PS/2 Mouse Interface (VHDL)
Debounce Logic Circuit (VHDL)

Contact

Comments, feedback, and questions can be sent to **eewiki@digikey.com**.

Logic Home: Table of Contents