Group 30
Gabriel Chootong 260 637 105
Sean Stappas 260 630 512

Report of the g30_keyboard_encoder Circuit

1. Circuit Description

- A description of the circuit's function, listing the inputs and outputs.
- Provide a pinout or symbol diagram.
- The VHDL description of the circuit

The purpose of this circuit is to encode a 64-bit signal into a 7-bit signal. The input is a signal "KEYS" representing 64 keys on a keyboard and the output "ASCII_CODE" is a 7-bit ASCII code. The circuit encodes all numbers as well as upper and lower case letters into their corresponding ASCII code. Figure X shows a pinout diagram of the circuit.

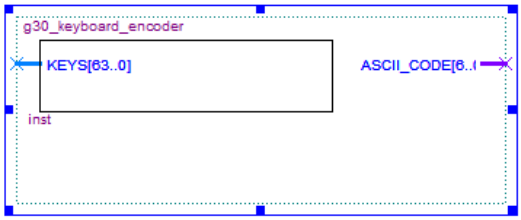

Figure X : Pinout Diagram of g30_keyboard_encoder Circuit

In order to encode the signal into its corresponding ASCII value, the circuit uses a 64-6 bit encoder. This 64-6 encoder is composed of four 16-4 bit encoders, and outputs the index number of the lowest significant bit of a 64-bit signal that is set to high. This component is therefore instantiated in the architecture of the circuit's VHDL code. Since each index value corresponds to a specific character, the circuit is programmed to output a specific 7-bit ASCII code depending on which bit is set to high.

In order to test the circuit, a test bench that generates every possible output value was designed. It uses a for-loop to set each bit of the "KEYS" signal to high one at a time. It is designed to wait for 10 ns in every iteration to make it possible to observe the output. Then, some random bit values are set to high in order to test for aleatory behavior.

Comment [SS1]: I already talk about this

	0	1	2	3	4	5	6	7
0	NUL	DLE	SP	0	@	P	`	p
1	SOH	DC1	!	1	A	Q	a	q
2	STX	DC2	"	2	B	R	b	r
3	ETX	DC3	#	3	C	S	c	s
4	EOT	DC4	$	4	D	T	d	t
5	ENQ	NAK	%	5	E	U	e	u
6	ACK	SYN	&	6	F	V	f	v
7	BEL	ETB	'	7	G	W	g	w
8	BS	CAN	(	8	H	X	h	x
9	HT	EM	)	9	I	Y	i	y
A	LF	SUB	*	:	J	Z	j	z
B	VT	ESC	+	;	K	[	k	{
C	FF	FS	,	<	L	\	l	|
D	CR	GS	-	=	M	]	m	}
E	SO	RS	.	>	N	^	n	~
F	SI	US	/	?	O	_	o	DEL

Figure 1. 7-Bit ASCII Table.

2. Testing

First, the keyboard encoder circuit was tested using a VHDL test bench. Figure 1 shows all 7-bit hexadecimal ASCII codes. For our design, we are only concerned with columns 2 to 5. So, when the least-significant '1' bit (LSB) of the input KEYS is at index 0, ASCII_CODE will have the value of the 'SP' character, i.e., 20_{16}. Similarly, if the '1' LSB is at index 1, ASCII_CODE will equal 21_{16}, the ASCII value of '!'. This goes on until the last value of column 5, $5F_{16}$ at character '_'. In our test bench, KEYS with a LSB '1' at index 0, 1, 2 … 63 are first tested (all 64 valid inputs), then the error case (no key is pressed), then some arbitrary input (multiple keys pressed simultaneously). This error case was our design choice, to keep track of when no key is pressed, and to have an initial state. Table 1 summarizes the expected output to the arbitrary input.

Table 1. Expected ASCII_CODE to Arbitrary Input

KEYS (unsigned value)	Least significant '1' index	ASCII_CODE (hexadecimal)
9223372071216611364	2	22
17870283321406128128	59	5B
2130303778816	36	44
1377042432	18	32
1378	1	21

Figure 2 shows the resulting waveform for each of the 'valid' inputs and the error case. These results confirm the expected behaviour, i.e., values of ASCII_CODE go from 20_{16} to $5F_{16}$, then 00_{16} at the error case. Figure 3 shows the resulting waveform for the arbitrary input. Once again, this confirms the expected behaviour, since the expected values in Table 1 appear in the waveform.

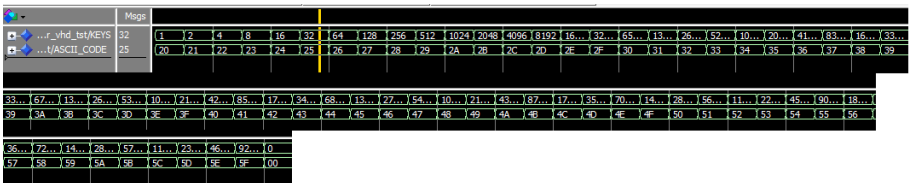

Figure 2. Valid Input and Error Condition Waveform.

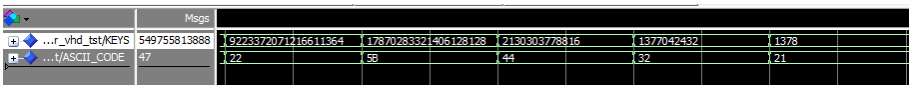

Figure 3. Arbitrary Input Waveform.

Afterwards, the keyboard encoder, as well as the LED decoder, were tested together on the Altera DE1 board. To do this, the g30_keyboard_encoder was combined with the g30_7_segment_decoder in the g30_keyboard_to_LED VHDL file. Since only 10 switches are available to us on the board, we decided to test all the numeric characters, from '0' to '9'. By the way we connected the pins to our circuit, the least significant character is associated with the least significant switch. So, if switch 0 is ON, character '0' should appear on the 7 segment LED. The behavior is also such that only the least significant character will be shown. For example, if switch 1 and 5 are ON, only '1' will be displayed on the LED. If no switches are on, the LED will be completely off. This port mapping to the Cyclone II EP2C20F484C7 can be seen in Figure 4.

Node Name	Direction	Location	I/O Bank	VREF Group	Fitter Location	I/O Standard	Reserved	Current Strength	Differential Pair
KEYS[0]	Input	PIN_L22	5	B5_N1	PIN_L22	3.3-V LV...default)		24mA (default)	
KEYS[1]	Input	PIN_L21	5	B5_N1	PIN_L21	3.3-V LV...default)		24mA (default)	
KEYS[2]	Input	PIN_M22	6	B6_N0	PIN_M22	3.3-V LV...default)		24mA (default)	
KEYS[3]	Input	PIN_V12	7	B7_N1	PIN_V12	3.3-V LV...default)		24mA (default)	
KEYS[4]	Input	PIN_W12	7	B7_N1	PIN_W12	3.3-V LV...default)		24mA (default)	
KEYS[5]	Input	PIN_U12	8	B8_N0	PIN_U12	3.3-V LV...default)		24mA (default)	
KEYS[6]	Input	PIN_U11	8	B8_N0	PIN_U11	3.3-V LV...default)		24mA (default)	
KEYS[7]	Input	PIN_M2	1	B1_N0	PIN_M2	3.3-V LV...default)		24mA (default)	
KEYS[8]	Input	PIN_M1	1	B1_N0	PIN_M1	3.3-V LV...default)		24mA (default)	
KEYS[9]	Input	PIN_L2	2	B2_N1	PIN_L2	3.3-V LV...default)		24mA (default)	
SEGMENTS[0]	Output	PIN_J2	2	B2_N1	PIN_J2	3.3-V LV...default)		24mA (default)	
SEGMENTS[1]	Output	PIN_J1	2	B2_N1	PIN_J1	3.3-V LV...default)		24mA (default)	
SEGMENTS[2]	Output	PIN_H2	2	B2_N1	PIN_H2	3.3-V LV...default)		24mA (default)	
SEGMENTS[3]	Output	PIN_H1	2	B2_N1	PIN_H1	3.3-V LV...default)		24mA (default)	
SEGMENTS[4]	Output	PIN_F2	2	B2_N1	PIN_F2	3.3-V LV...default)		24mA (default)	
SEGMENTS[5]	Output	PIN_F1	2	B2_N1	PIN_F1	3.3-V LV...default)		24mA (default)	
SEGMENTS[6]	Output	PIN_E2	2	B2_N1	PIN_E2	3.3-V LV...default)		24mA (default)	

Figure 4. Keyboard Encoder Pin Mapping On the Cyclone II EP2C20F484C7.

Figure 5 shows the resulting 7 segment LED patterns for all test inputs. This confirms the expected behaviour, showing all characters from '0' to '9', as well as the error case (no keys are pressed). Note that we used the least significant 7-segment decoder (on the right), and the other ones stayed on. Even though we did not test all the possible characters with the Altera DE1 board, we are confident that the circuit behaves correctly, since we also tested a big percentage of possible inputs in the VHDL test bench as well.

Figure 5. 7-Segment LED Output.

Grade Sheet for Lab #2

Fall 2016.

Group Number: 30 .

Group Member Name: Sean Stappas. Student Number: 260 639 512 .

Group Member Name: Gabriel Chootong. Student Number: 260 637 105 .

Marks			TA Signatures
2	1.	VHDL code for the *64:6 encoder* circuit	[illegible] .
2	2.	Simulation of the *64:6 encoder* circuit	[illegible] .
2	3.	VHDL code for the *keyboard encoder* circuit	[illegible] .
2	4.	Simulation of the *keyboard encoder* circuit	[illegible] .
2	5.	VHDL code for the *7_segment_decoder* circuit	AZ .
2	6.	Simulation of the *7_segment_decoder* circuit	AZ .
2	7.	Testing of the *7_segment_decoder* circuit on the DE1 board	AZ .

Each part should be demonstrated to one of the TAs who will then give a grade and sign the grade sheet. Grades for each part will be either 0, 1, or 2. A mark of 2 will be given if everything is done correctly. A grade of 1 will be given if there are significant problems, but an attempt was made. A grade of 0 will be given for parts that were not done at all, or for which there is no TA signature.