Two Tier Morse Code Encoder-Decoder

Tanmaey Gupta, Rushil Shah, Nidhin Harilal, Ribhu Vajpeyi

ES 203

Motivation

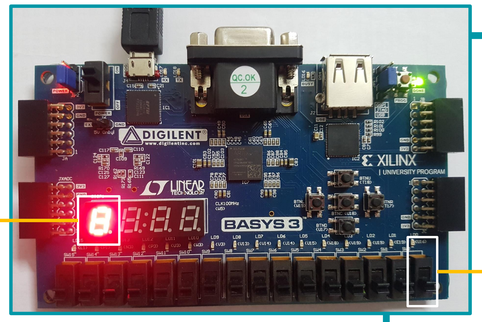
Morse code is a versatile method of communication which can be adapted to a wide variety of signals such as audio, visual, radio etc. It was designed in the USA by Samuel Finley Breese Morse in the 1830s. However It is still used today as an alternative in Aviation and Aeronautical communications fields since radio navigational aids such as VOR's and NDB's still identify the Morse Code. Morse Code has also been used as an alternative form of communication for people with disabilities in cases such as patients with full body paralysis have used the blinking of their eyes to communicate in the Morse Code. However, because of its simplicity, the Morse code can be easily decoded in between the process of transmission and receiving in applications such as long distance communications. Our project aims to combine the simplicity and adaptability of inputs the Morse code allows with the second level of encryption so that the code can be utilized in communications to make the communications secure.

Objective

Design a two tier real time encryption-decryption transmitter and receiver machine which takes in a dynamic input in the form of a Morse code and further encodes the message using the Base64 encryption algorithm.

Working for Encoding

User gives input on FPGA



DOT: 1 sec UP
DASH: 2 sec UP

READY for Input

DOT Input

DASH for Input

Algorithm

While taking input, all the characters are converted to their corresponding ASCII numbers following which all ASCII numbers are converted to their corresponding 8-bit binary number (Decimal to Binary number system conversion). Three 8-bit numbers are concatenated to form a 24-bit binary number which is then split into four 6-bit binary numbers. These 6-bit binary numbers are converted to their corresponding decimal number. This number is then looked up in a table consisting of 64 characters and the character corresponding to this number is given in the output. For example of an input of 1 character, 2 character output is given.

Base64 Encoding Table

Value	Char	Value	Char	Value	Char	Value	Char
0	A	16	Q	32	g	48	w
1	B	17	R	33	h	49	x
2	C	18	S	34	i	50	y
3	D	19	T	35	j	51	z
4	E	20	U	36	k	52	0
5	F	21	V	37	l	53	1
6	G	22	W	38	m	54	2
7	H	23	X	39	n	55	3
8	I	24	Y	40	o	56	4
9	J	25	Z	41	p	57	5
10	K	26	a	42	q	58	6
11	L	27	b	43	r	59	7
12	M	28	c	44	s	60	8
13	N	29	d	45	t	61	9
14	O	30	e	46	u	62	+
15	P	31	f	47	v	63	/

Binary Values of Encoded characters taken

Ru

BASE 64 Encryption

H

Algorithm

When each DOT/DASH input is being taken, the programme keeps track of the Morse encoding chart as well as the number of inputs. If 6 inputs are given or for 2 seconds no input is given after a certain set of inputs (<6) which matches a character, it converts it into a character. Each character is then converted into the 8-bit binary value of its decimal value. These binary values are taken for 1 character at a time and passed on for encoding.

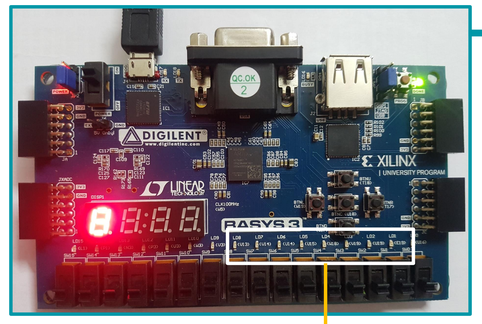
.-.. .-..

DOT/DASH get converted into letters

. -	A	- - - .	Q	- - - - .	g	- - - - -	w
- . . .	B	. - - .	R	- - - . .	h	- - - - - .	x
- - - .	C	. . .	S	- - - - -	i	- - - - - -	y
- . .	D	-	T	- - - - -	j	- - - - - - .	z
.	E	. . -	U	- - - - .	k	- - - - -	0
. . - .	F	. . . -	V	- - - - - .	l	- - - - -	1
- - .	G	. - -	W	- - - - - .	m	- - - - -	2
. . . .	H	- . . -	X	- - - - - .	n	- - - - -	3
. .	I	- . - -	Y	- - - - - -	o	- - - - -	4
. - - -	J	- - - .	Z	- - - - - .	p	- - - - -	5
- - -	K	- - - -	a	- - - - - .	q	- - - - -	6
. - . .	L	- - - - - .	b	- - - - -	r	- - - - -	7
- -	M	- - - - - .	c	- - - - -	s	- - - - -	8
- .	N	- - - - -	d	- - - - - .	t	- - - - -	9
- - -	O	- - - - - -	e	- - - - -	u	- - - - - -	space
. - - .	P	- - - - - .	f	- - - - - .	v	- - - - - -	end char

Encoded output seen on LEDs of FPGA

010001110



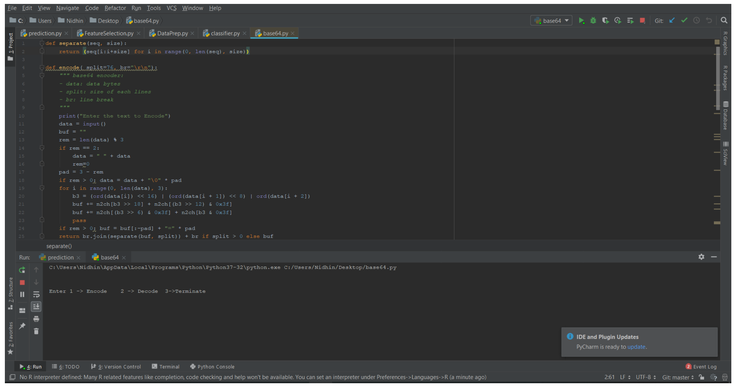
8 bit output shown on 8 LEDs for each encoded character.

Working of Decoder

For decoding, the user inputs the ASCII values of the encoded sequence of characters to the decoder. Computation is then performed on this input string and the message is decrypted. This decrypted message is then shown on the FPGA's LEDs

Validating the Encoded Message

To validate the encrypted or decrypted string, we have created a Python User Interface which takes in its input the encrypted or decrypted string and computes the corresponding decrypted or encrypted string. This User Interface helps the user to check whether the 2 tier FPGA encoder/decoder is computing the output accurately.



References

- https://en.wikipedia.org/wiki/Base64
- https://www.base64decode.org/
- https://en.wikipedia.org/wiki/Morse_code