Custom IP Cores For Encryption

By Jason Agron

This document describes how to interact with a custom IP core that implements the *Caesar Cipher*. The Caesar Cipher is a simple encryption process based on the substitution cipher in which every letter in a plaintext string is replaced with another letter that is located at a fixed offset from the original letter. The basic process of the Caesar Cipher is the following:

- For a given encryption offset, d.
- Read in the original plaintext letter and convert it to ASCII, x.
- The encrypted letter will be in ASCII will be (x+d).
- To decrypt the letter, just subtract the encryption offset.

In this lab the Caesar Cipher IP core will be provided to you in the form of a user_logic file. In order to use this file, just replace the original user_logic file of your custom IP core with the new file and re-build the hardware platform. This will replace the internal logic of the IP core with the logic for the new encryption core.

The encryption IP core is structured as follows:

- MODE register:
  - Located at base + 0x0.
  - 32-bit, Readable/Writeable.
  - Configures the mode of the encryption core.
  - The MSB is the configuration bit.
    - 1 = Encryption, 0 = Decryption.
      - Encryption mode = 0x80000000.
      - Decryption mode = 0x00000000.
- INPUT register:
  - Located at base + 0x4.
  - 32-bit, Readable/Writeable.
  - Contains the input character to encrypt/decrypt.
- OFFSET register:
  - Located at base + 0x08.
  - 32-bit, Readable/Writeable.
  - Contains the encryption offset to be used in encryption/decryption.
- RESULT register:
  - Located at base + 0x0C.
  - 32-bit, Read-only.
  - Contains the result of encryption/decryption.

Basic Outline of Project Steps:

1) Replace the user logic file of your custom IP core with the encryption core user logic file.
2) Create a software driver for the encryption IP core to perform encryption on a NULL-terminated string.

Hardware Modifications:

To replace the user logic file of an IP core. First open up a file browser and go to the location of your XPS project. There should be a directory called pcores/, open it up. From here you should see another directory that is the name of your custom IP core, open it up too. From here there should be a file called user_logic.vhd under hdl/vhdl/. This is the file that you need to replace with the special encryption core file.

Software Modifications:

On the software side, you must write code to do the following:

1) Print the original NULL-terminated string to the screen.
2) Encrypt the string.
3) Display the encrypted string.
4) Decrypt the string.
5) Display the decrypted string (should be the same as the original string).

Software Hints:

How to display a NULL-terminated string:

```
#define MAX_SIZE 20
int main()
{
	int counter = 0;
	char originalMessage[MAX_SIZE] = “fpga’s jungle books”;

	// Initialize counter
	counter = 0;

	// Loop through each character until a NULL character is found
	while( originalMessage[counter] != (char)NULL )
	{
		xil_printf(“Character = %c \r\n”, originalMessage[counter]);
		counter = counter + 1;
	}
	return 0;
}
```