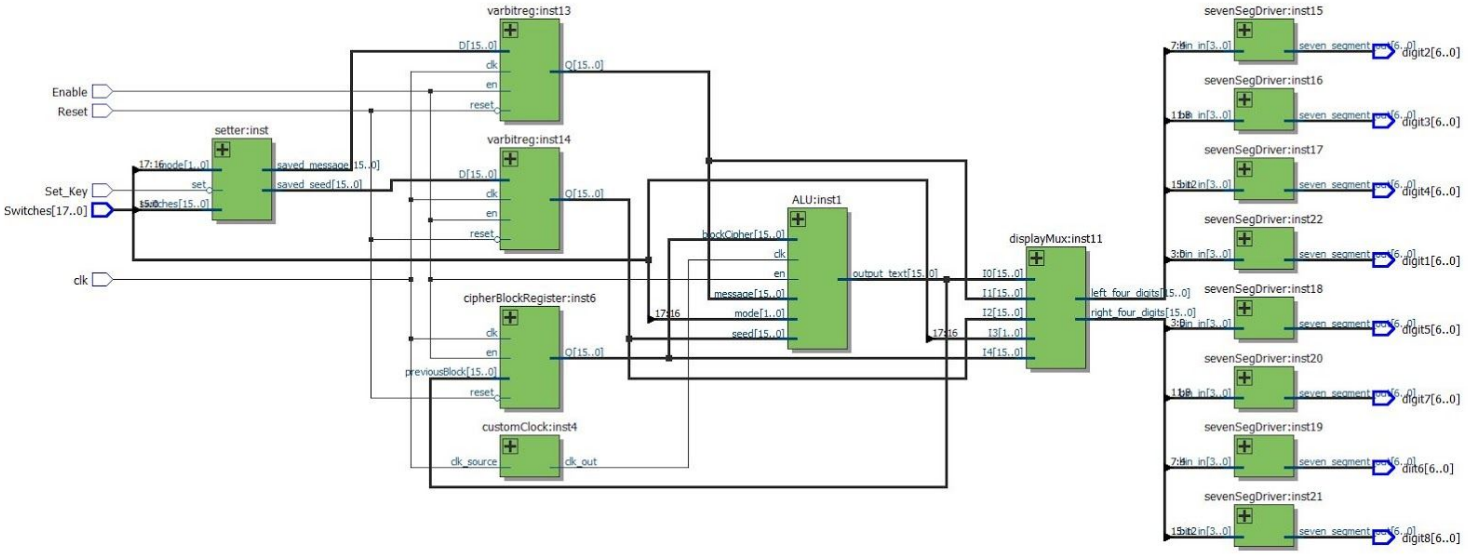
ECE 2277A - Semester Project
Official Readme Document

Name: Jake Raw, 250798333
Date: December 14th/2018



RTL View of the VHDL Application

Project summary:

Using the VHDL programming language this application can encrypt and decrypt 4-bit hexadecimal messages using a strategy of Linear Feedback Shift Registers for pseudo random number generation. The application is designed to run on the Altera DE2-115

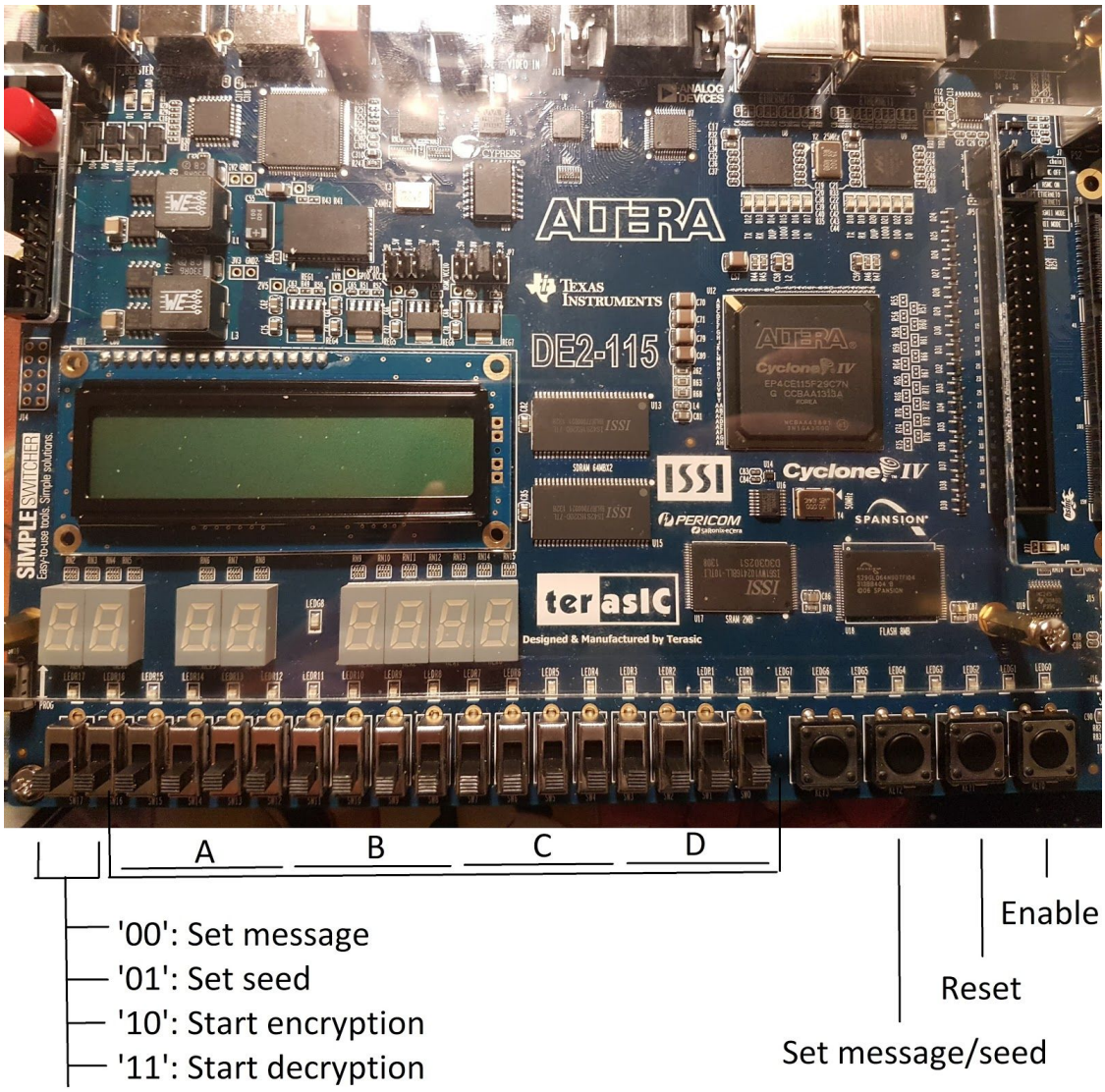
A user can input any of the four possible hexadecimal characters A-F using 16 of the board switches. The user has access to two input modes: setting a message or setting a seed value. These modes can be toggled with a switch, and both values can be saved into unique registers. These values can be indefinitely set, cleared and accessed.

When both a seed and message value are set the user can switch to the Encryption or Decryption mode, treating the entered message as either a plaintext or ciphertext respectively.

The Algorithm Logic Unit uses its own clock that rises approximately every second (as opposed to the default Altera board clock speed), triggering either a generation of an encryption block cipher or a decryption block cipher each second. In encryption mode, the right half of the 8 digit seven-segment display the originally entered messaged with the constantly updating block cipher value on the left half. In decryption mode, the originally entered 4-bit seed value is displayed on the left half of the 8 digit seven-segment display and the constantly updated plain text on the right.

The overall goal is that a user can input a 4-bit message, count how many block cipher iterations are made in the seven-segment display encryption mode, then expect the same amount of iterations in reverse to decrypt the cipher text into the original message.

Controls Overview:



High Level Board Overview of Instructions

Detailed Breakdown of controls:

Switches[17..16]: Select the modes of operation.

- ‘00’:

  View the current value of the message in Hex[3..0] LED displays
  The left 4 LEDS, Hex[7..4] will display ‘0000’

- ‘01’:

  View the current seed value in the Hex[7..4] LED displays
  The right 4 LEDS, Hex[3..0] will display ‘0000’

- ‘10’:

  The 4 left LEDS, Hex[7..4] will display the constantly updating ciphertext (encrypted)
  The right four, Hex[3..0] will only display the message set before the encryption

- ‘11’: The right four, Hex[3..0] will display the constantly updating plaintest (deciphered)
  The 4 left LEDS, Hex[7..4] will only display the seed value set before the encryption

Switches[15..0]: 16-bit space for entering 4 Hexadecimal BCD values

- Switches[3..0]: The least significant bit, in BCD Hexadecimal
- Switches [7..4]: The second least significant bit, in BCD Hexadecimal
- Switches [11..8]: The second most significant bit, in BCD Hexadecimal
- Switches [15..12]: The most significant bit, in BCD Hexadecimal

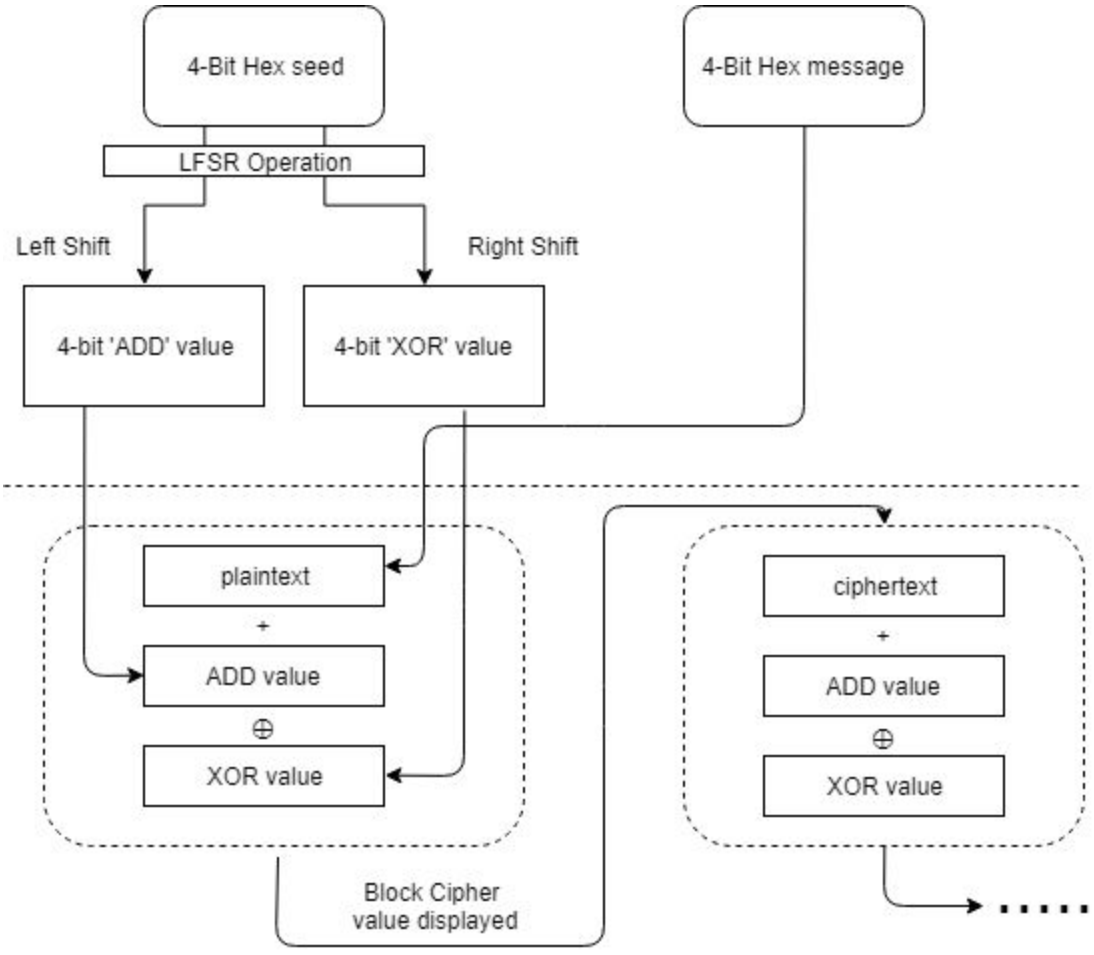
Keys[2..0]: Persistent functions

- Key[0]: When pushed, disables all registers from updating
  When unpushed, enables all registers.

- Key[1]: When pushed, sets all registers values to '0..0'
  When unpushed, there is no function.

- Key[2]: If the board is in '00' mode (set message), when pushed, will save the currently entered value into the message register
  If the board is in '01' mode (set seed), when pushed, will save the currently entered value into seed register.

Encryption/Decryption Strategy:



Overview of the CBC mode block ciphering

This encryption models simple Cipher Block Chaining (CBC) encryption. Based on the value of the seed value entered two pseudo random values are determined with a Linear Shift Feedback Register operation.

- Using LSFR right shift on the seed:
  A random 4-bit value is created that is added onto the previous cipher text for each block

- Using LSFR left shift on the seed:
  A random 4-bit value is created that is XOR'd with each current block cipher

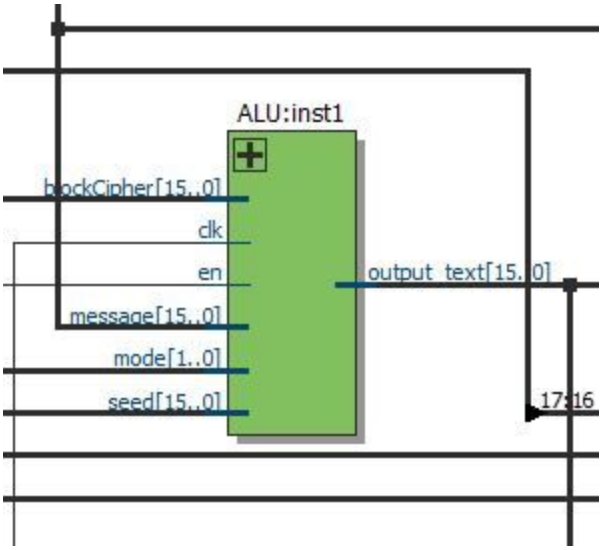
These two pseudo random values are used in tandem to consistently scramble the original message. Since the seed length is 4-bits there will be 2^4 uniqure block ciphers before a roll over and values will be repeated.

The decryption algorithm works almost identically except the order of the ADD value and XOR values are reversed, before a ciphertext/plaintext is output.

Component Breakdown:

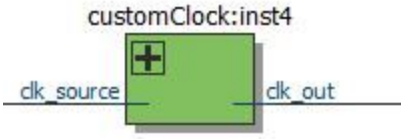
This section will outline each of the components in the RTL diagram view.

Algorithm Logic Unit



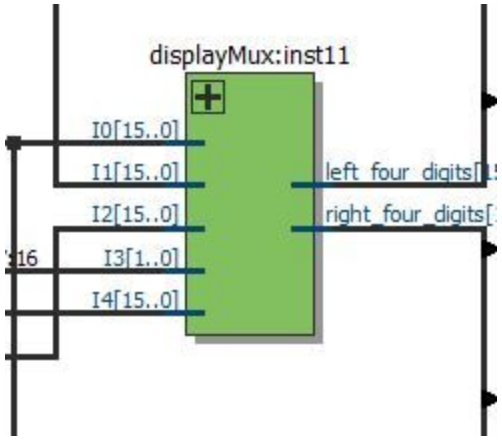
The algorithm logic unit performs the encryption and decryption operations. The ALU is enabled when the encryption or decryption modes are set. Every rising edge from the ALU's clost 1 second clock will perform either a encryption or decryption block cipher and send the value to the block cipher register and the display multiplexer for a live update in the Hex LEDs.

ALU Custom Clock



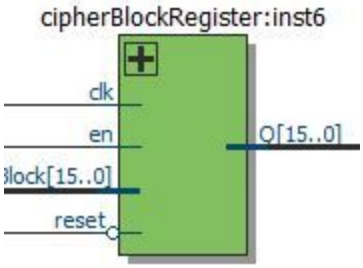
The ALU custom clock is designed to trigger an event approximately every 1 second, using the Altera board's clock as a reference.

Display Multiplexer



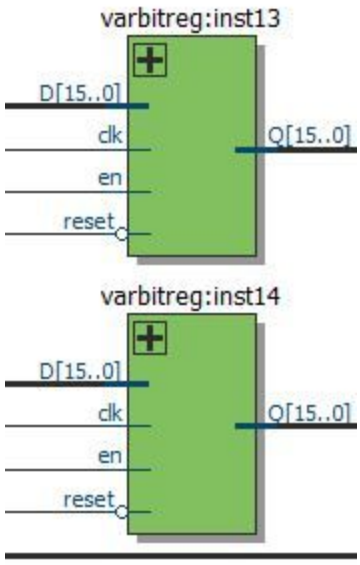
The display multiplexer has 4 different outputs that it can display to the Hex LEDs. Option 1: display the message value concatenated with ‘0000’ (set message). Option 2: Display the seed value concatenated with ‘0000’ (set seed). Option 2: Display the live block cipher updates on the left with the original message on the right (encryption). Option 4: Display the live block cipher updates on the right with the original seed on the left.

Block Cipher Register



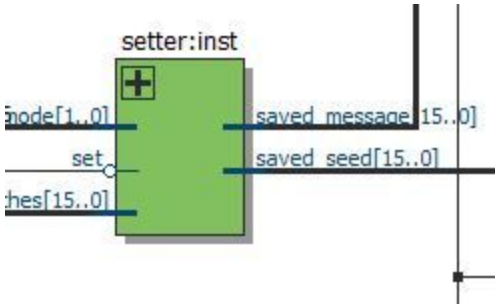
Each block cipher needs to be saved in order for the ALU to calculate the next block cipher. The block cipher register saves each of these states to be used by the ALU.

Variable Bit Registers: Message and Seed



The message and seed values are stored in generic D Flip Flops. They can be set, cleared and accessed by the display multiplexer or ALU at any time.

Message and Seed Input Setter



The setter is a simple multiplexer that saves the currently entered value on the switches to either the seed register or message register depending on the current mode.

Seven Segment Display Drivers

These simple drivers convert BCD Hexadecimal numbers into formats for the Altera board's LEDs.