COL215: Mini Project Completion Report

Student_1: Pavas Goyal (2018CS10363)

Student_2: Dipen Kumar (2018CS50098)

Project Title: Text Encryption/Decryption (Crypto-Box)

Proposed Plan and Deviation from this-

1. We proposed to use simple stream cipher to encrypt/decrypt data before storing in memory module.
- ➢ We ended up doing both stream cipher before storing in memory and finally block cipher for transmitting final output.
2. We initially didn't planned to use seven segment display in ours mini project for encrypting/decrypting.
- ➢ Now we are using seven segment display to show the key in hexa-decimal entered by the user.

Achievements-

We have achieved our goal for encrypting and decrypting text using both simple stream cipher and block cipher scheme. Features of our Crypto-Box are mentioned below-

1. We have used 8 slide switches for key to be entered by the user. Description of the key follows. We have used this key to generate two numbers- 'a' and 'b'. This 8 bits key is broken down to 4 bits two different key. First one be 'a' and second be 'b'. This 4 bits vectors are being converted into integers. Hence in sort, the key given by the user is used to generate two integers independent of each other.
2. We kept reading the text feed by the user on gtk-term serially through serial receiver module. This signal is then given to module when signaled by timing module. The timing module signal receiver to give parallel input to memory when rx_reg is equal to full.
3. There exist a private_1 module between receiver and memory which does a simple stream cipher so that the data stored in memory is not the original one rather that this data is encrypted using a default in-built key of the system design. This ensure the privacy of the data when memory data of the memory is revealed.

4. Similarly, There is a module called public_1 module which provides an interface between memory and transmitter. This basically produces a series of random number taking ‘a’ and ‘b’ as seeding parameters. This module when gets signal from timing module when transmit push button is pressed starts getting data-address (index of the array where this data was stored). With the help of this index number and two constants- ‘a’ and ‘b’ created from user key, we create another final key as follows using cantor pairing hash function. And pass this key to transmitter.
5. Finally transmitter receives data from memory and final key from public_1. It xor these two keys to get a final encrypted output, which is further displayed.
6. Final key is kind of pseudo random number generated as the parameter index number of address array is kept on changing.

Inputs to our Design-

1. Text/file on gtk-term
2. 8 slide-switches (user entered key)
3. 2 slide-switches (operating mode)

Outputs to our Design-

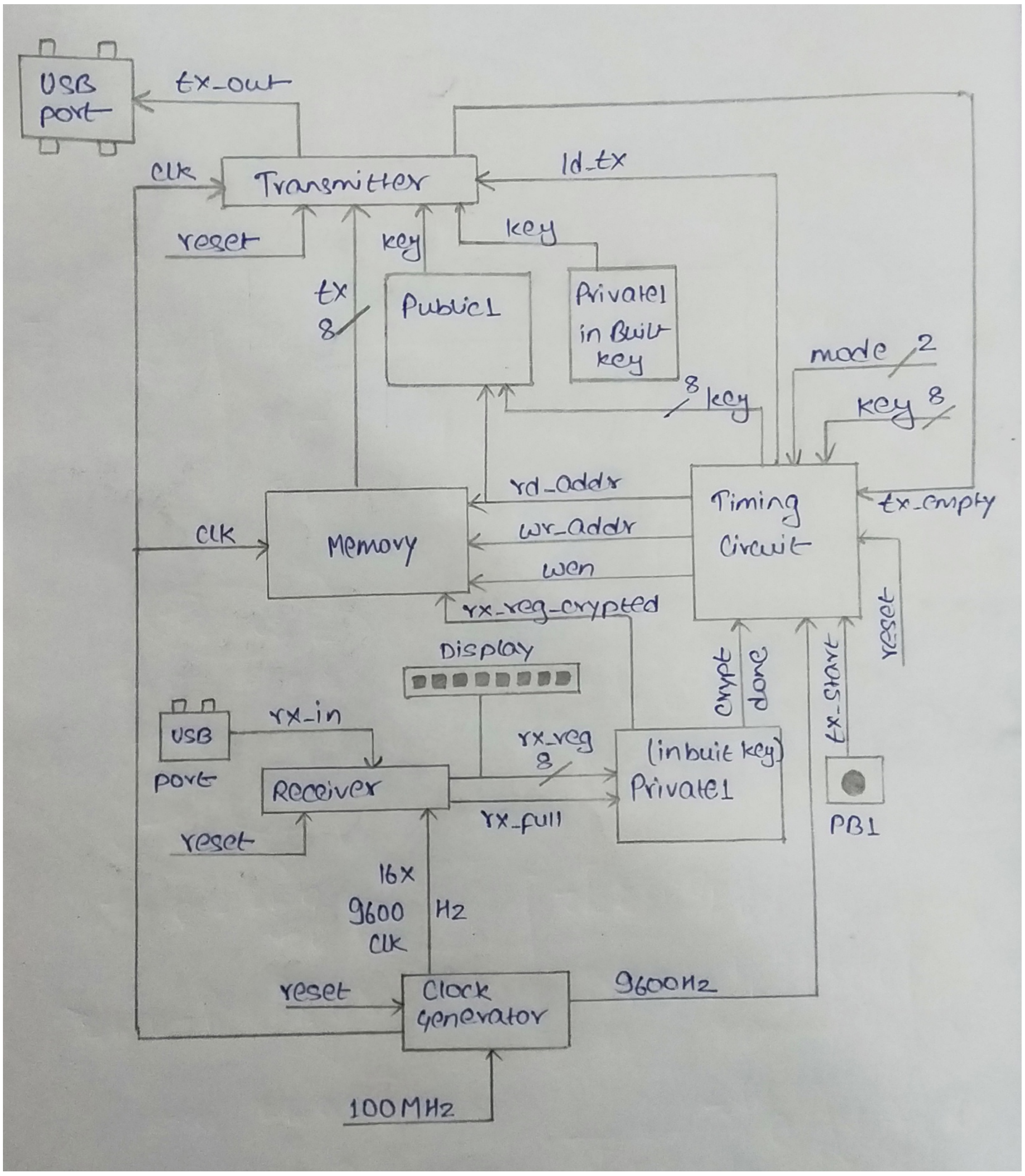
1. Encrypted/Decrypted text/file on gtk-term
2. 8 LEDs (received signal) + 8 LEDs (transmitted signal)
3. Seven-segment display (Hexa-decimal form of key

Problems Faced-

1. We were getting right results when we were giving input text by typing through keyboard on gtk-term but was getting wrong result when we were using raw file as an input. Bug hidden was that while typing we are giving input character by character at slow rate i.e. inter-arrival time between two character read by our receiver was sufficiently large enough to make its state back to ideal from start/stop. But while reading input from raw file it was very fast. Raw file give input to gtk-term with a rate which was faster for our receiver to get back to ideal state. Hence not receiving the input signal properly and hence not giving right encrypted output which when further decrypted was giving something else what was input.

2. Hash function used for generating pseudo random number sequence with three input ‘a’, ‘b’ and position number (index/counter) was giving very high value when ‘a’ and ‘b’ were set large number, which further resulted in some kind of malfunction of key generated. In order to ensure that ‘a’ and ‘b’ are sufficiently small in the range we assigned it to a positive integer value whose binary form length is at max 4, i.e. 4 bit std_logic_vector.

Final Block Diagram-



Project Calendar-

- Week11- We proposed our plan, ideas, and target to achieve. We did theoretical work on our project design, its block diagram, functionality of different modules. Input output areas and so on.
- Week12- We started working on our ideas, created modules on VHDL. We verified our ideas in the domain of project with prof. Anshul Kumar. We modified our ideas according to it. We added extra features. We searched for a powerful hash function which distribute over large value more or less uniformly with little collisions.
- Week13- We started working on our new modified idea (verified by professor) and started implementing those. We also worked on cantor pairing hash-function implementation.
- Week14- We completed our encrypting/decrypting process, hash function. We started adding one user friendly feature of showing the key entered by user in hexa-decimal form which was entered in binary form through slide switches on Basys-3 board. We started working on our final report and submission process and completed it. We gave demo of our project.