REFLEX TESTER

-Hardhik Donapati (2016cs10339)
-Vinayak Rastogi (2016cs10345)

Problem Statement:

This project aims to test how fast you can respond after seeing a visual stimulus or rather hand-eye coordination. There would be a “reset” button to reset all the states and make the system ready for the next reflex test. It will then display a “hi” message on 7-segment display to show that it is ready for next input. The user needs to press the “start” button to start a random number generator that turns on the “output led” after a random interval. (Randomness is important here, otherwise the time of led on can be memorized). After the LED is on, the timer starts and counts the time till the user presses “stop” button. At this time, the reaction time should be displayed on the 7-segment display and hi message should be removed from 7-segment display.

Approach:

Our task basically is to test the user’s reflexes by calculating the time taken by them to react to an external stimulus and then outputting the time taken. The user gets to decide when he/she is ready for the next round by pressing a button.

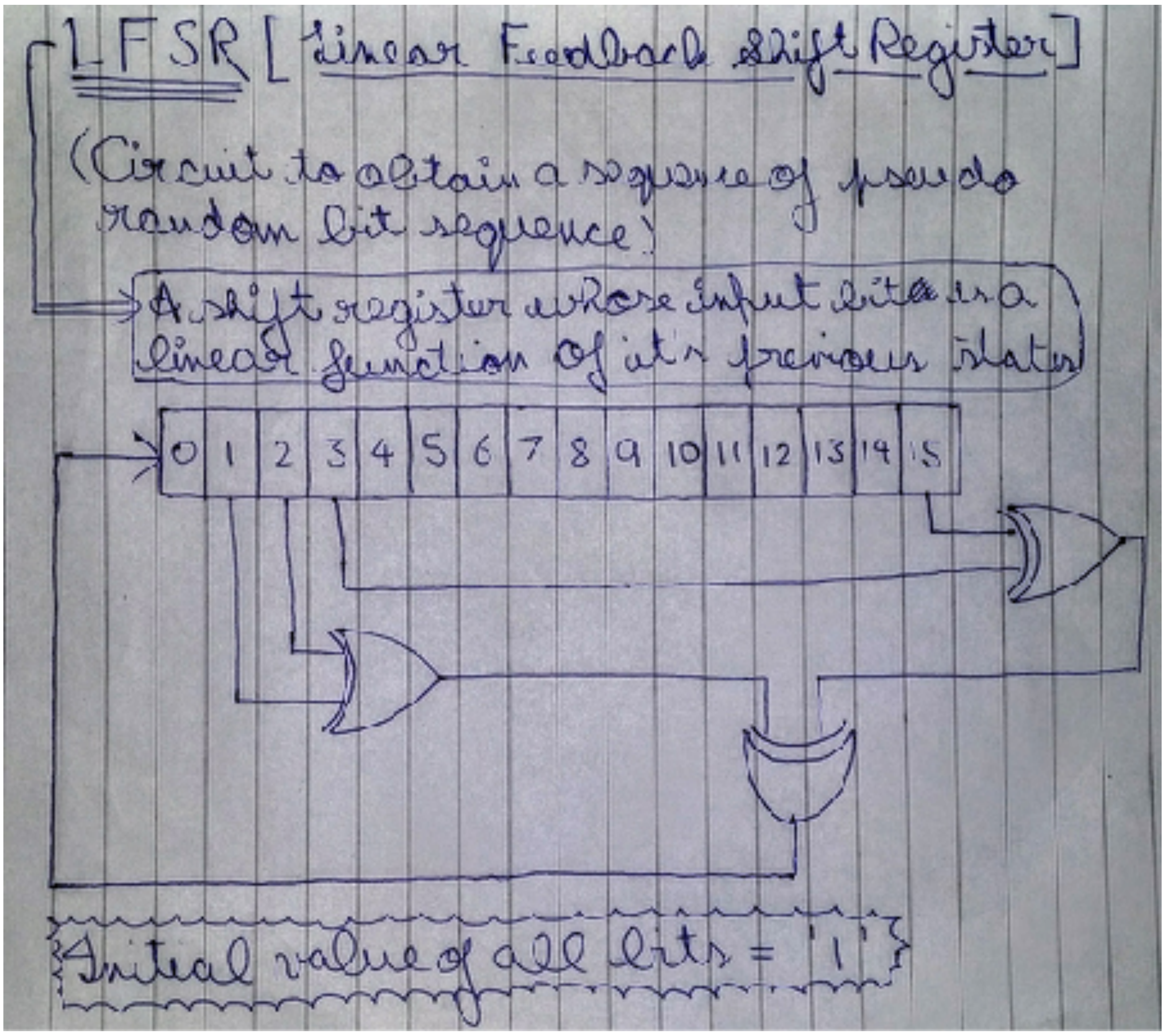
The stimuli here is an LED light that switches on to which the user’s response would be the pressing of a particular button. The task will be performed on an FPGA (BASYS) board.
The user will be greeted with a “HI” message on the 7seg. display and by pressing another particular button the user decides when he’s ready.

The “Hi” disappears and after a random (pseudo) time interval the LED light (the stimulus) gets lit to which the user is supposed to respond.
A clock will be initiated when the light is on and will stop when the response by the user is received. Then, the time taken for the user’s reaction

will be outputted on the Seven Segment Display on the board and the process is repeated once again when the user pushes the button that gets the board ready for the next test.
The crux of the Reflex Tester is to generate a random sequence of numbers, which can be used as time intervals between the switching on of the output LED. A sequence of pseudo random bits can be obtained using shift registers and a XOR gate. A block of a particular length of this sequence can be used to generate the desired random numbers.

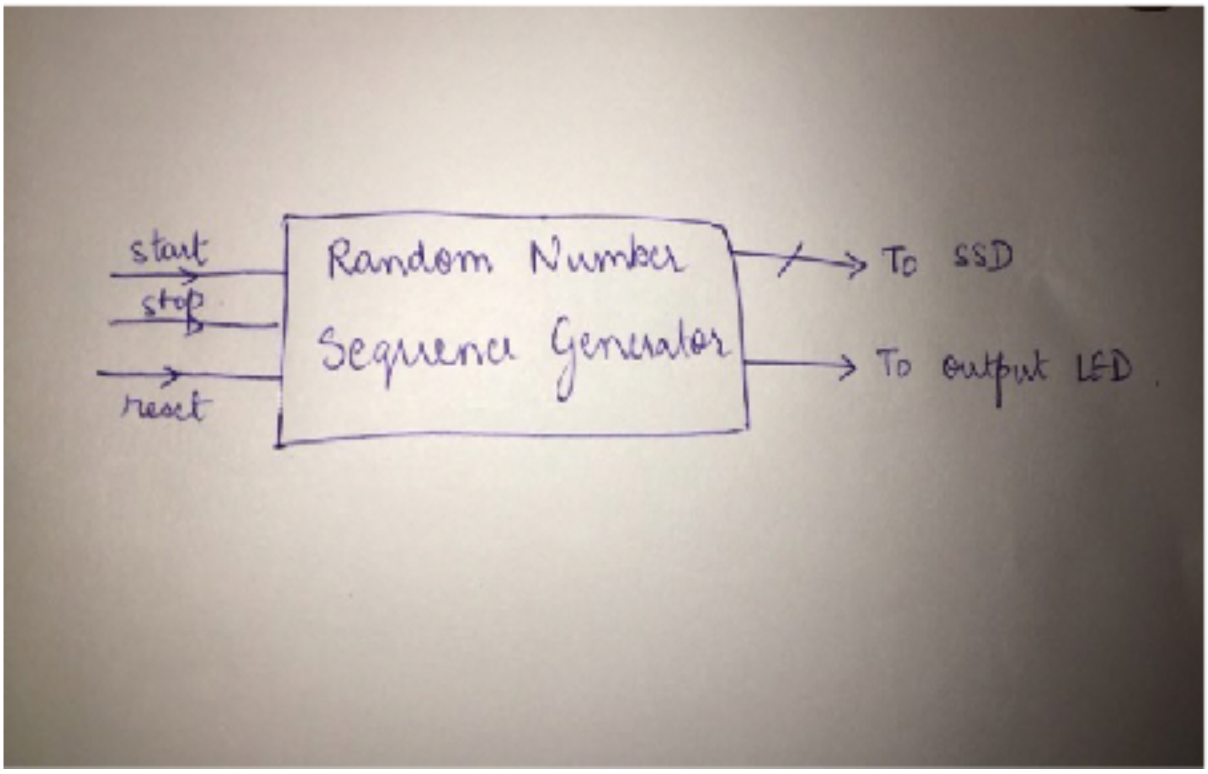
Block Diagrams

Pseudo Random number generator:


LFSR [Linear Feedback Shift Register]
(Circuit to obtain a sequence of pseudo random bit sequence)
A shift register whose input bit is a linear function of it's previous state
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
Initial value of all bits = '1'

The block diagram contains Random Number Generator with Three inputs: start, stop and reset switches and
Two outputs: an output bus to the Seven Segment Display and an output port

to the Output LED.



Validation Methodology

We didn’t make a Test-bench file because it wouldn’t have made any sense. The point of the project was testing the reflexes of a human user after reacting to a randomly timed signal. The Testbench would have violated the whole point of that.
On the unit level, the only thing that COULD be tested using a testbench file, was the D-Flip-Flop, which is too basic to do and absolutely trivial.
Our main Strategy for test and Demonstration was to keep checking all the components (dividing/breaking the whole problem into small modules/ functions and working on one module at a time) with all the required and appropriate test cases, individually for each step. And once we had the basic complete prototype(with all the working functionalities that are required as specified in the problem statement), we generated the bitstream file and ran the program on the FPGA board. After running once and making sure that the modules combined in a bugfree manner, we giave our fellow group students a chance to play and fiddle with the board ,see the project in action for themselves and also tell us what could we improve upon.

Further (Extra, not replacement) functionalities/modofications (not necessarily specified in the problem statement and not violating it either) were also be added as per wish.

Detailed design and Implementation Details

We implement a state diagram with three states. The first state is entered on pressing start or reset buttons. The signals are initialized here, a message “0HI0” appears on the SSD and the system moves onto the second state. Here a pseudo random bit sequence is utilized to create random time intervals between glowing of output LED. The time interval between the LED glow and button press response is added to a counter. The third state is attained when the end button is pressed, and the average reaction time is displayed on the SSD.
The second state utilizes 16 d- flip flops and for generate statements to create a Linear Shift Resistor. We use xor of four bits 1, 2, 3 and 15 to create pseudo random sequence with maximum N.
The top module of the project is named miniproject_ReflexTester. It includes two component instantiations of D-flip flop and seven segment display. We used the seven segment display implemented by us lab4 with appropriate changes.
‘ToSSD’ signal represents the input to the SSD module. A slow clock signal ‘slowc’ aids in reducing the frequency of appearance of random bits in the bit sequence.
The coding is complete, synthesis shows three errors which have to be rectified and testing will be directly done on the FPGA board.

Assumptions

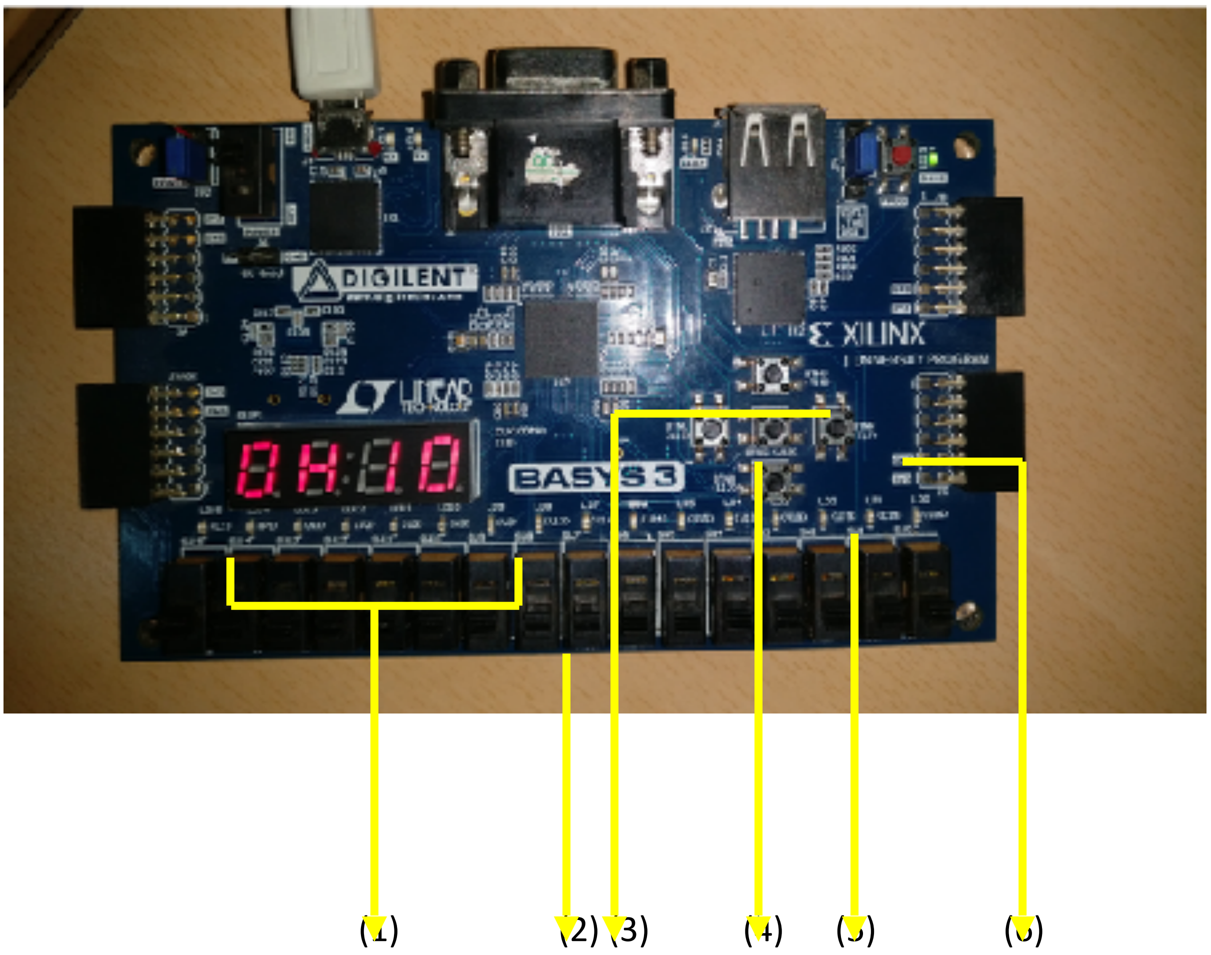
We assume that the user does not press the user_input continuously, and presses the button only once on seeing the LED glow. The final output is the ‘average’ reaction time to a visual stimulus.

Additional Theory

We utilize a Pseudo Random Sequence of bits generated using Linear Feedback Shift Register. It is ‘random’ in a sense that the value of an element of the sequence is independent of the values of any of the other elements. It is

'pseudo' because it is deterministic and after N elements it starts to repeat itself, unlike real random sequences. Here N is n*(2^n-1). And the n we implement is 4.
The average reaction time of a human to a reaction stimulus is 0.25s. So we choose to light up the output_LED around every 4-7s.

BOARD:



1. -> SEVEN SEGMENT DISPLAY
2. -> LED LIGHT (STIMULUS)
3. -> START
4. -> RESET
5. -> STOP
6. -> USER_INPUT

BIBLIOGRAPHY:

https://pdfs.semanticscholar.org/42e7/490ec8905ea8afe618c6882f2b050ece2ae4.pdf (pseudo random generation)