FRACTAL LANDSCAPE GENERATOR

KESHAV JANYANI & ABHISEK SAHU
ROLL NO. 160260005, 160260024

B.Tech, 2^{nd} year,
Dept. Of Physics, IIT BOMBAY

Course Project report for
EP 230: Electronics Lab III
Taught by *Prof. Pradeep Sarin*

1 Project Summary

The aim was to implement a recursive algorithm named Diamond-Square Algorithm on FPGA. The Algorithm is used to create natural looking terrains. Our code calculates the z-coordinate at each point of a predefined grid. We have implemented the code on a grid of 9×9.

Algorithm:

1. Initialize corner points of grid. We have used a counter to implement this part. A counter from 0 to 255 runs on the clock of circuit. Whenever we press Reset,the value stored in counter is stored in the corner points.
2. Square Step: Store the sum of average of the corner points and a random number in the center grid point.
3. Diamond Step: Consider the four possible diamonds having endpoints at the values calculated and store the sum of average of endpoints and a random number at the center of the diamond.
4. Reduce the range of random numbers.
5. Make four smaller blocks and repeat the steps till you cover all points.

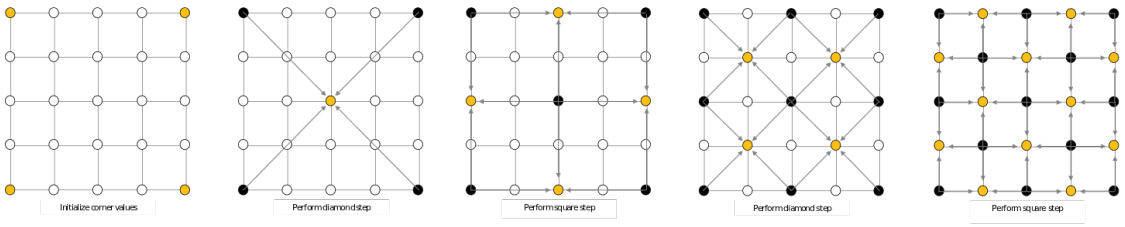


Figure 1: Diagram depicting the square and diamond algorithm

2 Design block Diagram

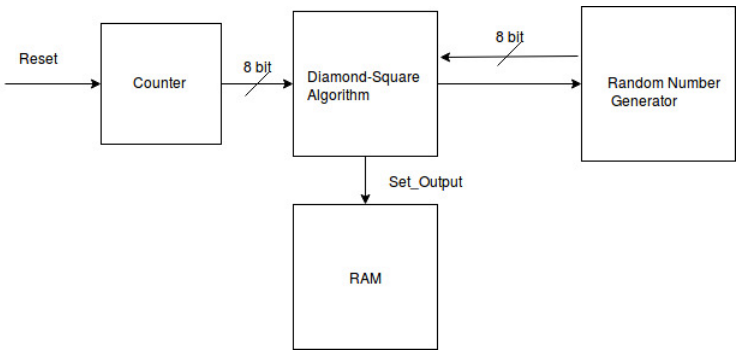


Figure 2: Block Diagram displaying structure of code

3 Main problems faced and how to overcome

1. **Displaying Output of Algorithm:** We needed to display the output in a good looking way. We thought of using VGA but to display the 3D data on VGA, we would have had to use colour maps, which might not have looked beautiful on the small scale we are implementing the algorithm. So we decided to send the data to a computer and use python to display the output in a better way.

2. **Sending Data to Computer:** To do this part, we first tried to implement UART, but soon realized that this would take too much time and effort on our part. We found a shortcut by using RAM to send data. We used Quartus Prime's in system memory content editor to see the content on RAM. We modified our code such that the output is stored to RAM instead.

3. **Creating Random Numbers:**We needed to create a lot of random numbers in our algorithm. The ideas we had were:

   (a) Using a resistor and measuring it's thermal noise. (Rejected because we would have required to used a ADC and other complications involved with it)

   (b) Using a huge counter and pressing stop and using the counter value at that instant as the random number. (Rejected because it was impractical to press a button 77 times for the algorithm to be completed)

   (c) Using a linear shift registor. We initialized a 8 bit number. we xored its 5^{th} and 7^{th} bit and placed it in the 0^{th} bit after left shifting the number by one bit. We did this each time we needed a random number in the algorithm.

4 Special hardware parts used

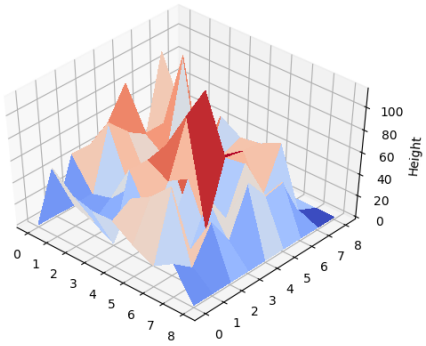
1. **Laptop:** We used a PC and a python code to read Hex Files and plot a 3d surface plot for the images we have got.

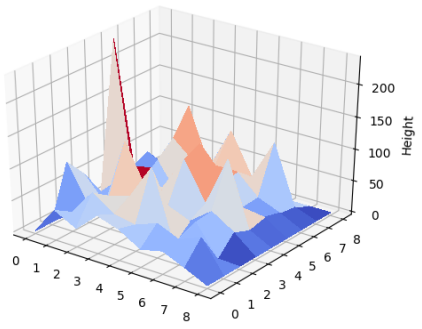
5 Use of FPGA in implementing this project

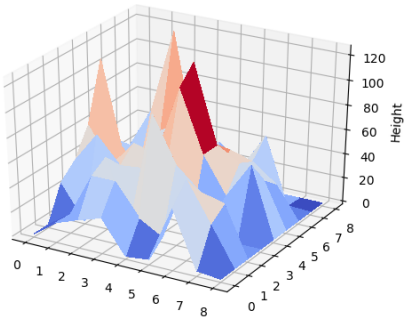
The algorithm we are implementing is recursive and as it goes deeper into the implementation, a lots of blocks can run independently of each other. If such an algorithm is implemented on a microprocessor, it would have sequentially gone through each block, hence increasing the execution time. The use of FPGA is justified because the algorithm now executes the blocks that can run independently, in a parallel manner.

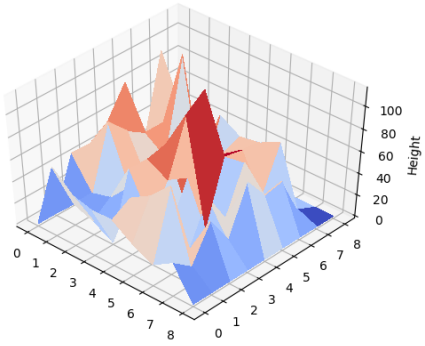
6 Results

Here our some of the outputs of the implementation of our algorithm:









7 References and Important Links

- Our Inspiration :
  `https://people.ece.cornell.edu/land/courses/ece5760/FinalProjects/f2010/tss64_rs423/rs423_tss64/hldesign.html`
- Our Code:
  `https://github.com/kjanyani/FPGA_Fractal_Generator`
- Wikipedia page for Diamond - square algorithm:
  `https://en.wikipedia.org/wiki/Diamond-square_algorithm`