ECE 5760: Lab 1

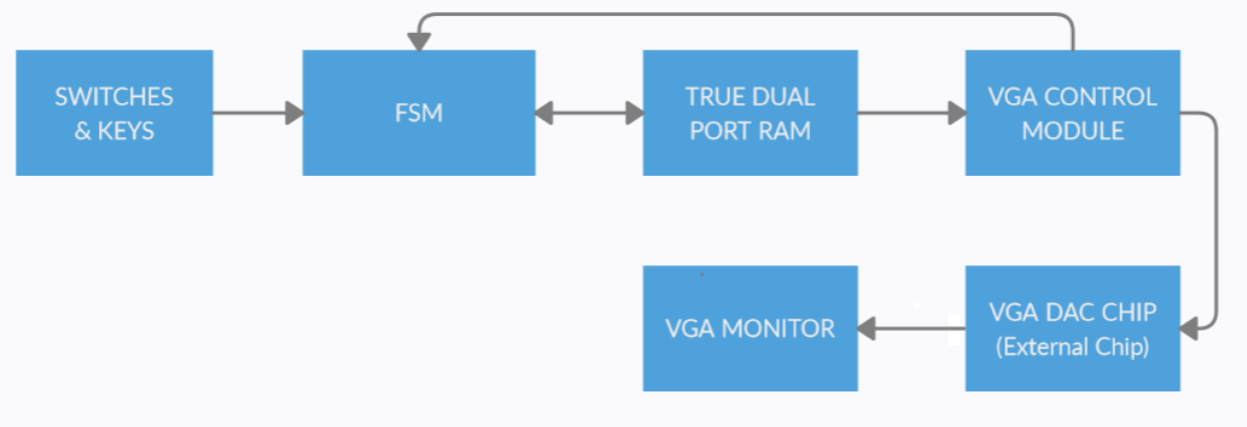
February 8, 2021

One-Dimensional Cellular Automata

emre.demirli@metu.edu.tr

Introduction

One dimensional Cellular Automata is implemented on a DE2 FPGA board. The results are placed on a True Dual-Port Ram so that a VGA monitor can display it. The design included a VGA Control Unit, a Reset Unit, a PLL, a True Dual Port RAM, Keys/Switches, and a finite state machine. The design is capable of creating one-dimensional cellular automata with the rule ranging from 0 to 255. An overall system diagram is shown below.



Hardware

True Dual-Port RAM and PLL are created by the Quartus Megawizard tool. VGA Control Module and Reset Module are taken from the lab's examples page. The monitor that is used is LG Flatron F700B. No other external hardware is connected to the FPGA.

ECE 5760: Lab 1

February 8, 2021

One-Dimensional Cellular Automata
emre.demirli@metu.edu.tr

Finite State Machine

The rows are implemented cyclic to deal with the edge cases of the cellular automata. There are a total of 28 states. Each state either do a very simple task or decides the next state. The number of states is definitely reducible, but the FSM is designed in consideration of debugging and limited time.

States

reset: Frame Buffer is cleared, counters and flags are initialized, and the rule is registered. Also, the next state is decided by reading SW[17].

middle_point_0: A white point placed in the middle of the first line.

random_line: The state repeats itself 640 times, and with each iteration, consecutive random points are placed to the first line of the screen.

read_x: Address of the upper pixels of interest is put to the address bus of the RAM. Six consecutive read states are required because the RAM inputs and outputs are registered.

read_x_y_boundary: Implementation of cyclic rows is achieved by these states. They are only reached when the calculation comes to the edges of the screen.

SW_x_state: The consecutive three values read by read_x states lead to corresponding SW_x_state. In this state, the rule decided by the switches is written to the RAM.

decide: This state decides whether there is a left limit condition or not.

stop_FSM: The variables and flags are reset. The FSM halts in this state until KEY[3] is pressed. Then, the copy operation starts.

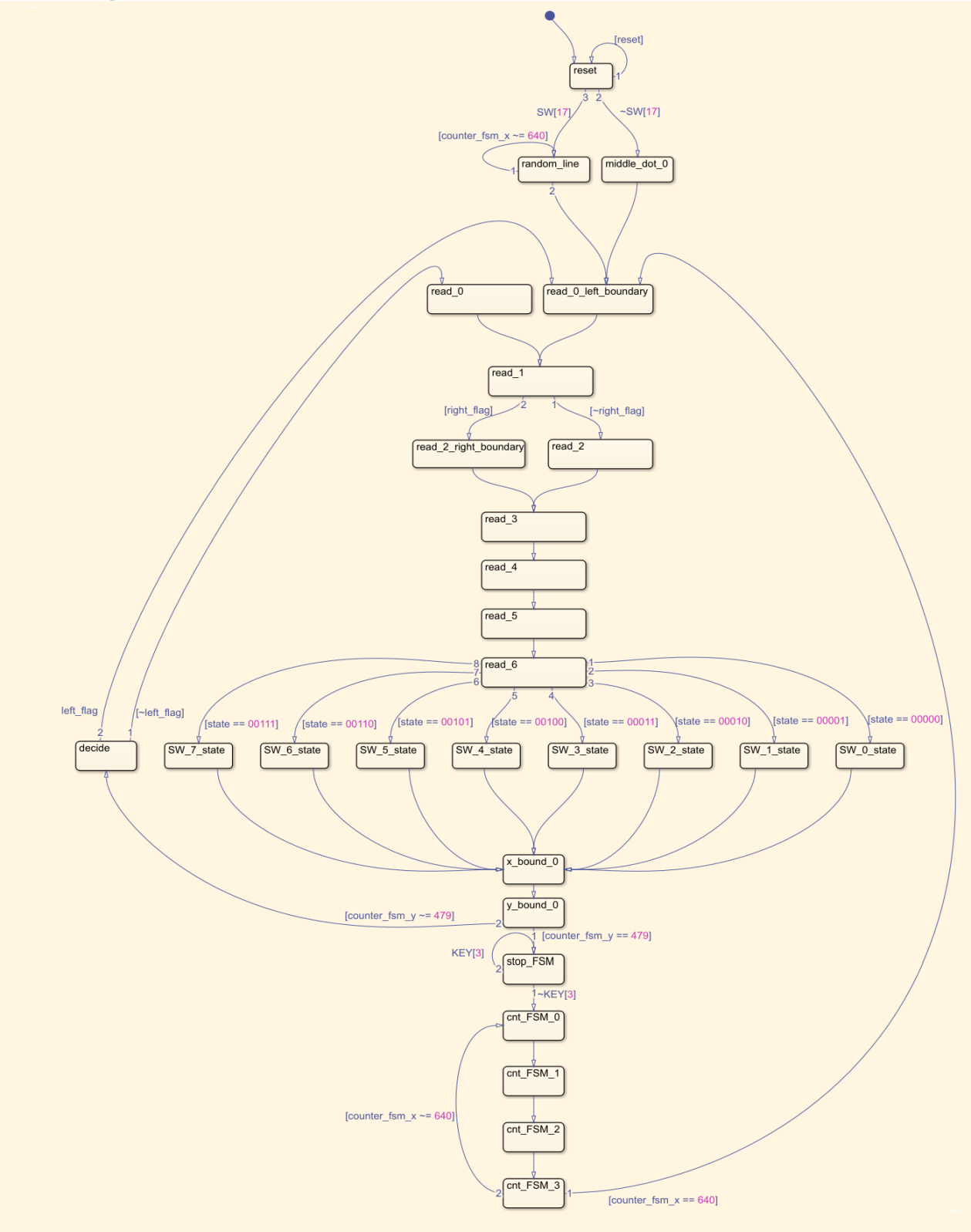
cnt_FSM_x: These states read the last row of the screen and write it back to the zeroth row so that the patterns can continue to evolve.

ECE 5760: Lab 1

February 8, 2021

One-Dimensional Cellular Automata

emre.demirli@metu.edu.tr



This is not a functional MATLAB state chart. MATLAB is used solely because drawing charts with it is easy.

ECE 5760: Lab 1

February 8, 2021

One-Dimensional Cellular Automata
emre.demirli@metu.edu.tr

Testing and Debugging

Initial tests are done by different test benches to verify that every module is functioning. When simulating the overall design, the VGA Control Module is excluded from the design because of the initial incrementation delays of the Coord_X and the Coord_Y. These delays were increasing the overall time of the simulation. Thus, after verifying that the module is working, the module is replaced by simple counters. The rest of the simulation is carried on by eye inspection of RAM. Since the only possible flaws can occur either at the edges or in the middle. Verifying the correctness of the edge and the middle bits led to a working design.

Possible Design Improvements and Current Flaws

Since I have a limited amount of holiday, I can not spend more than few days for my each project. Therefore, implementing these improvements left as an exercise to the reader.

- When KEY[3] is pressed, the design is supposed to write the step n+1 to the first line of the screen, not the step n. Unfortunately, I noticed this when I was writing the lab report. A simple solution to this is expanding the memory by one row and incrementing the limits of if statements by one. Then, the invisible precalculated 480^{th} line will be copied to the zeroth line as intended.

- There are a lot of inefficiencies in the FSM. To improve it, you can easily separate the reading and writing part of the FSM and create a shift register for the reads. So that a new pixel can be calculated every cycle. As a similar and simple example, if you increase the number of write states of cnt_FSM_x to 3 and increment the counter in each cnt_FSM_x, you can copy a pixel in every two rather than four cycles.

- The outputs of the RAM does not have to be registered. In fact, they were not registered in the lab examples. In addition to the previous improvements

removing the output registers will lead to less Logic Element usage. Also, data input to the VGA Control Module is also registered again by a register at the negedge of the clock. Which is added another unncessary overhead to the circuit.

- The write operation of the reset state is unnecessary. Altera documentation states that M4K blocks are always powered up to zero, which is what we wanted. And after the power up, there is no need for a clear screen operation because the screen is filled with the new design faster than the human eye can catch.

- There is a setup timing violation between VGA Control Module and Reset Module. But since the reset signal comes when the VGA is in sync interval, it is okay to leave it like that.

Conclusion

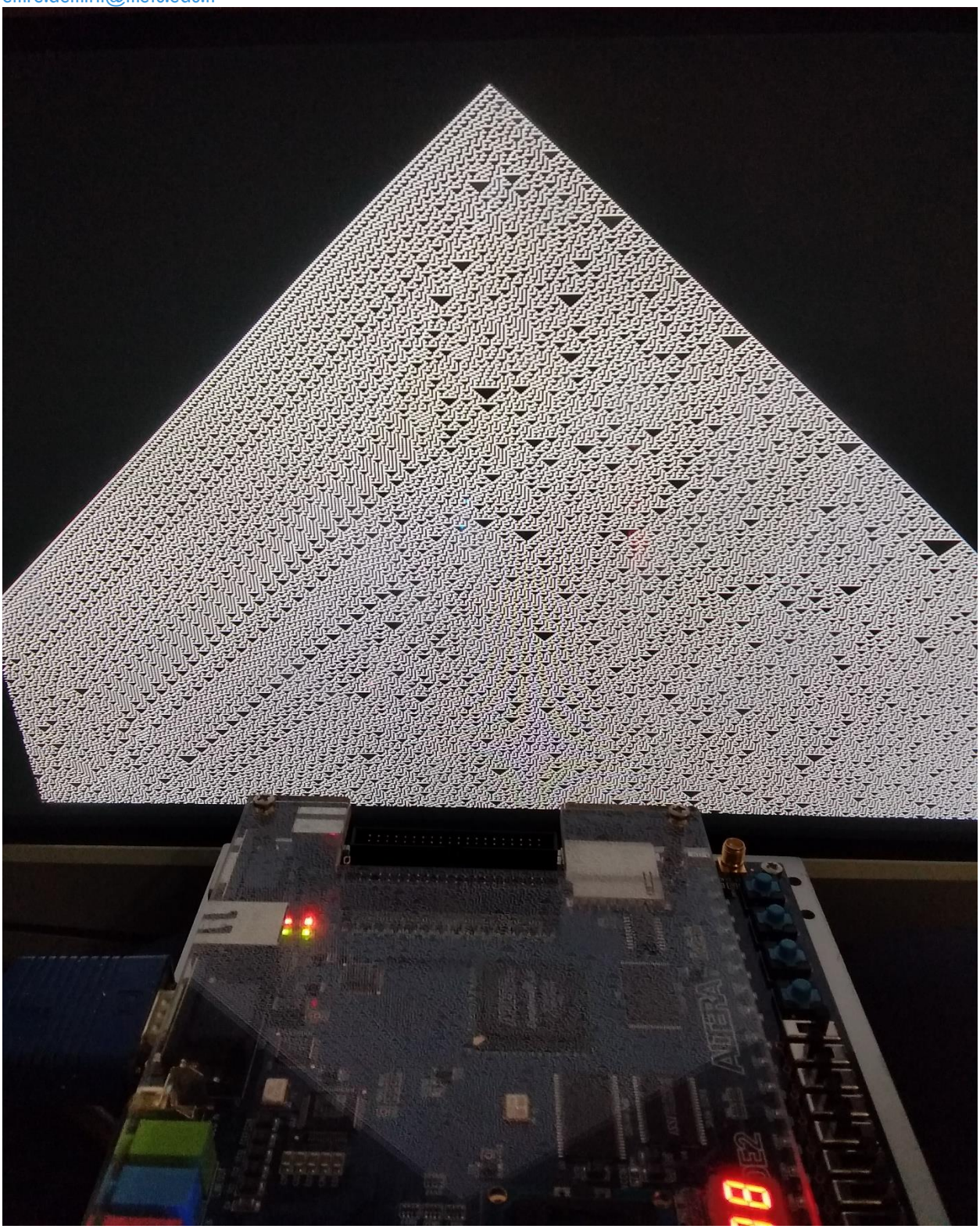
This lab was my first proper FPGA project. I 'experienced' a lot of new stuff. Including pain from mixing sequential statements (they apparently were not) even though I have coded a CPU on HDL before, happiness of not seeing any red lines on Modelsim, and the frustration of forgetting that I registered the outputs. I wish I had more time to improve my project. But I will probably try the lab two when I have it.

ECE 5760: Lab 1

February 8, 2021

One-Dimensional Cellular Automata

emre.demirli@metu.edu.tr



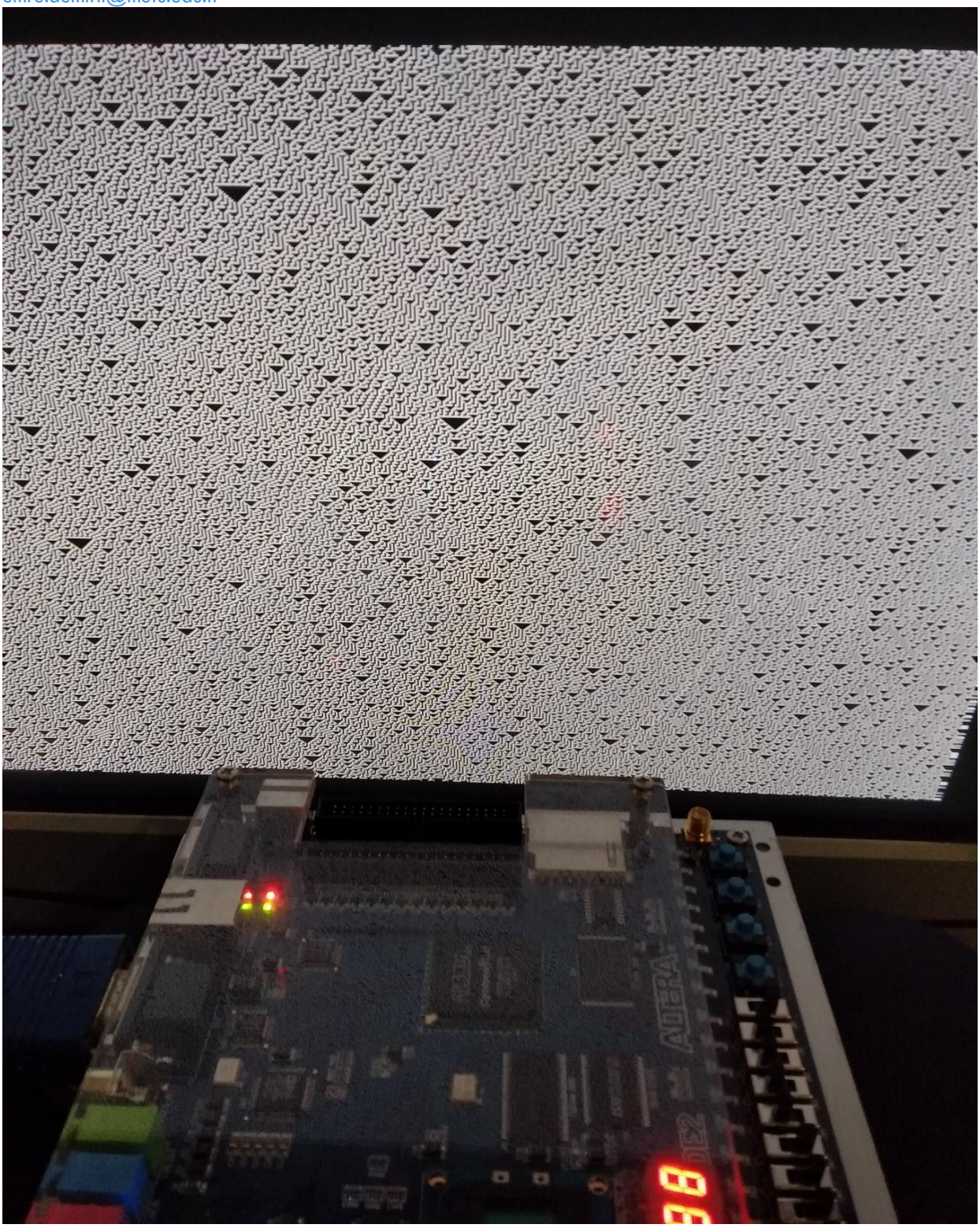
RULE-30

ECE 5760: Lab 1

February 8, 2021

One-Dimensional Cellular Automata

emre.demirli@metu.edu.tr



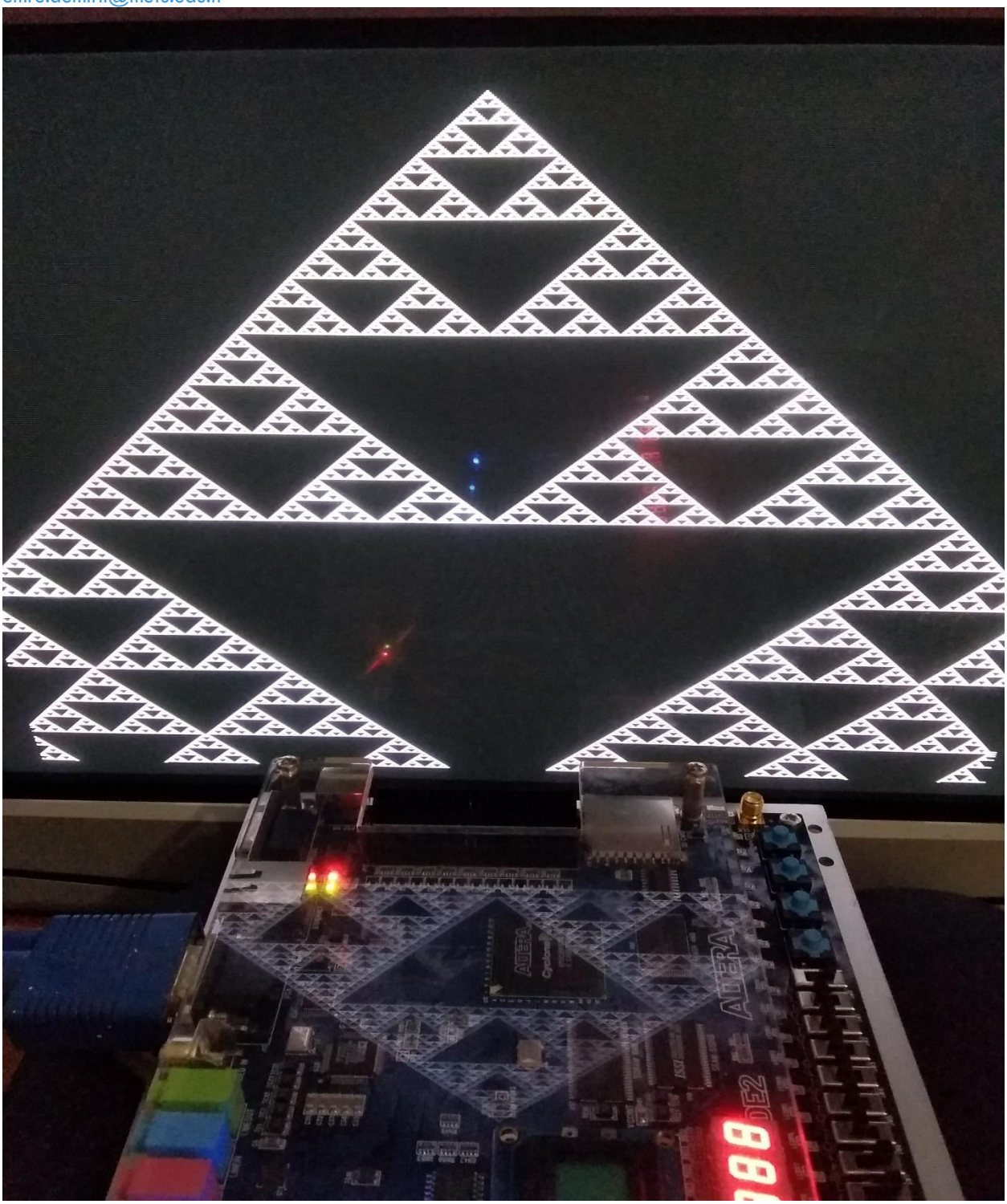
RULE-30 Continued

ECE 5760: Lab 1

February 8, 2021

One-Dimensional Cellular Automata

emre.demirli@metu.edu.tr



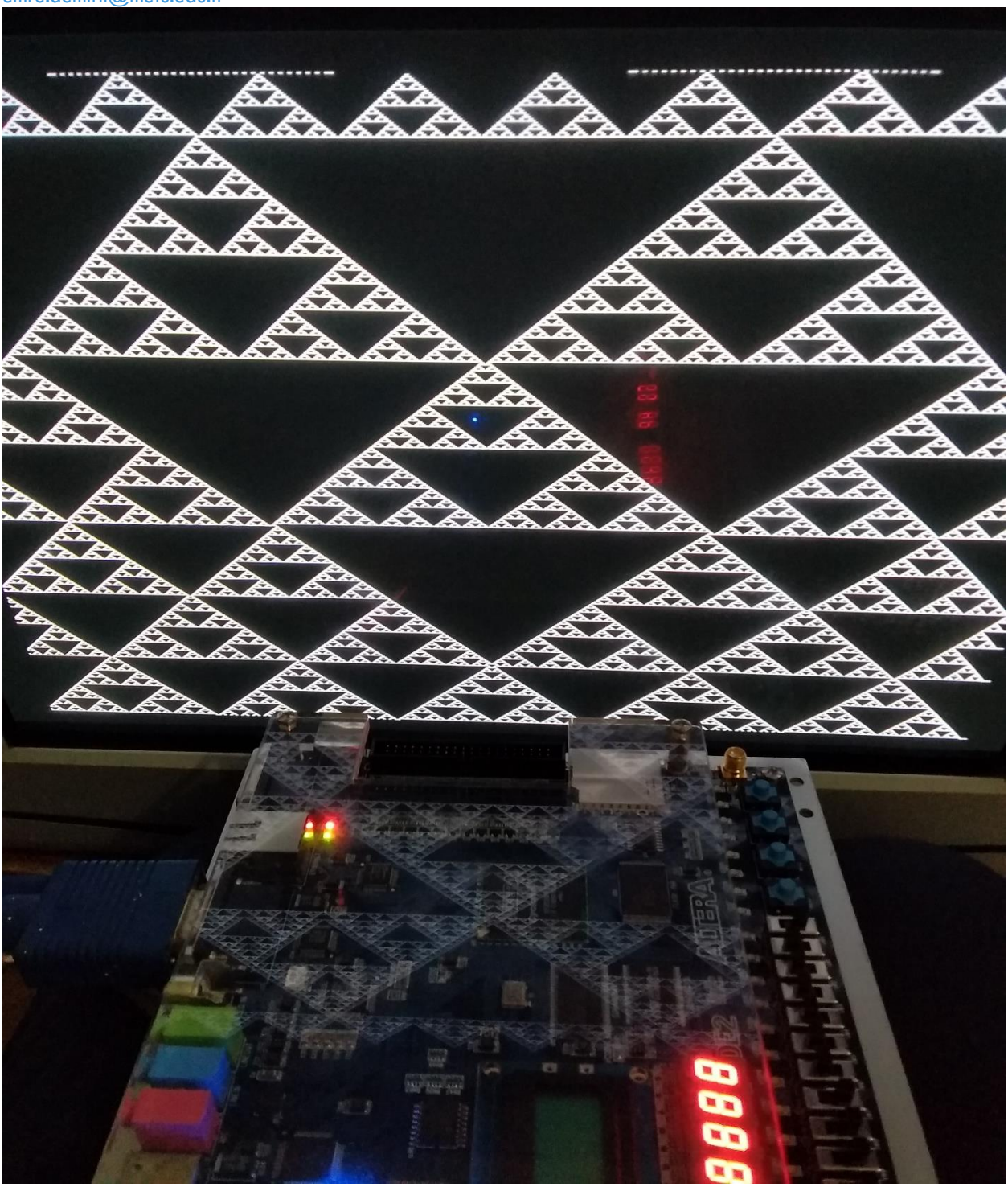
RULE-126

ECE 5760: Lab 1

February 8, 2021

One-Dimensional Cellular Automata

emre.demirli@metu.edu.tr



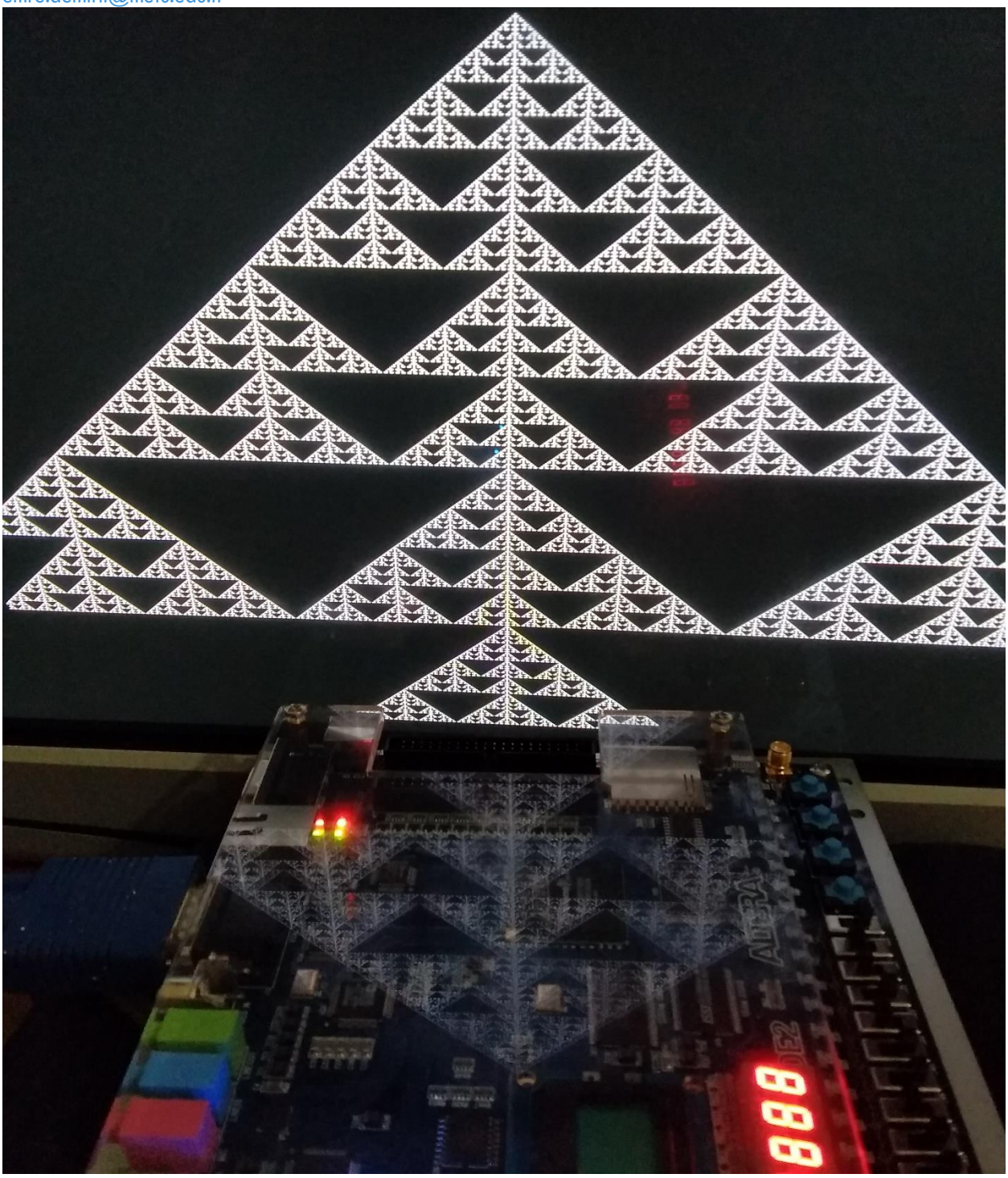
RULE-126 Continued

ECE 5760: Lab 1

February 8, 2021

One-Dimensional Cellular Automata

emre.demirli@metu.edu.tr



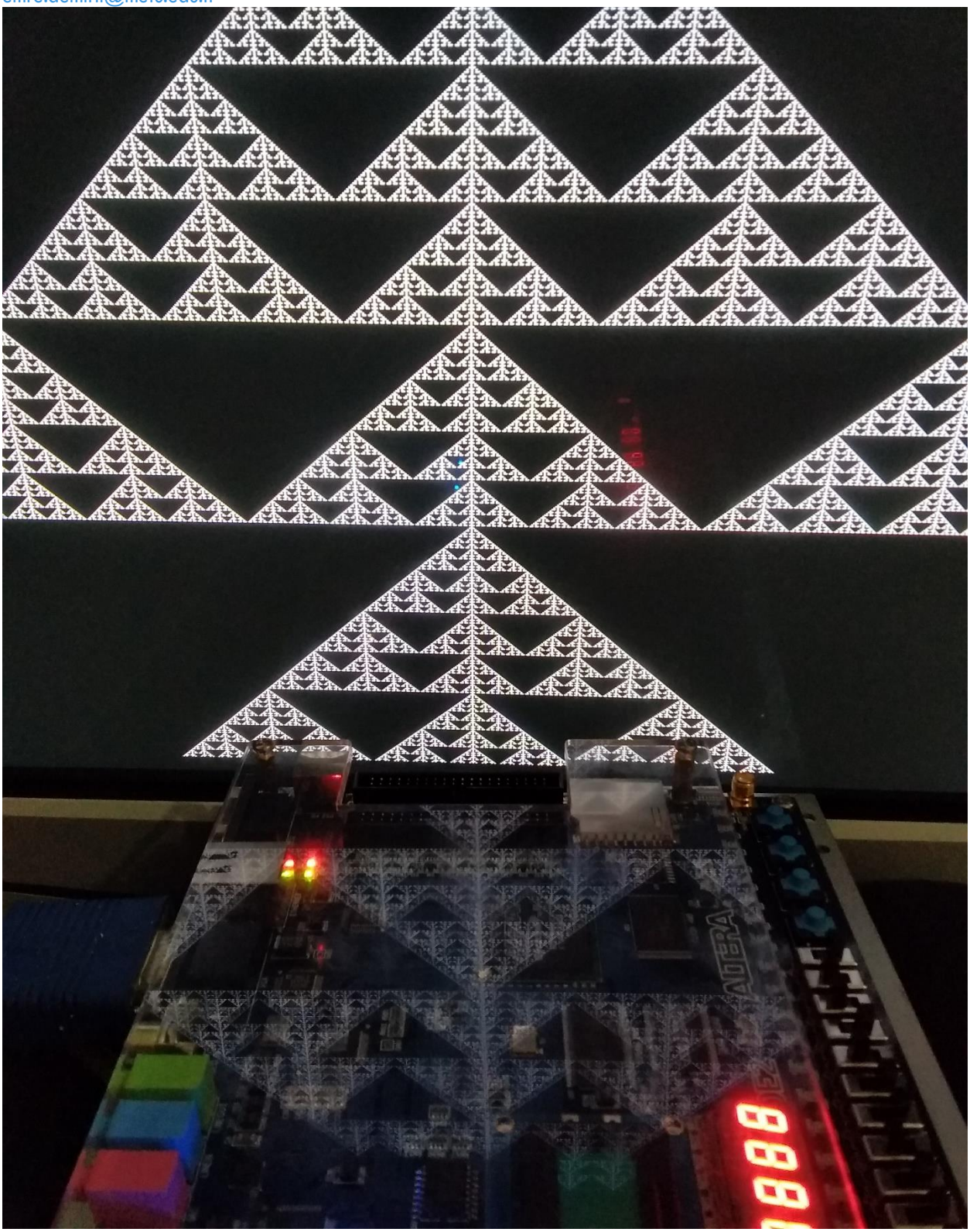
RULE-150

ECE 5760: Lab 1

February 8, 2021

One-Dimensional Cellular Automata

emre.demirli@metu.edu.tr



RULE-150 Continued