ECE 5760: Laboratory 1

2-dimensional Game of Life.

Introduction.

In this assignment you will build a device using the FPGA and HPS which runs a state machine to compute and display a Cellular automaton (CA) on a VGA monitor. The operation of the CA solver for Conway's Game of Life will be controlled by interaction with the USB mouse attached to the HPS.

Procedures.

1. Read Linux on DE1-SoC
2. Read University Program DE1-SoC_Computer_15_1
    1. When you load the unmodified sof file of the *better chopped down system* onto the fpga:
    2. download the zip
    3. open the project
    4. open the programmer and click `start`
3. You are going to be programming C to run the Conway CA state machine and mouse.
   Therefore you are expected to be able to program in Linux/GCC. Read about:
    1. /dev/mem for i/o mapping, e.g. simtec and local
    2. Debian Linux. We are using a Debian variant, I believe.
    3. GCC on Linux, e.g. die.net
    4. USB on DE1-SoC Note that the demo mouse code on this page is blocking.
       You will probably need to add code to make the device non-blocking.

```
5. //needed for nonblocking mouse read()
6. int flags = fcntl(fd, F_GETFL, 0);
   fcntl(fd, F_SETFL, flags | O_NONBLOCK);
```

4. You are going to connect the VGA controller using Qsys tools. Read about:
    1. Qsys
    2. VGA on DE1-SoC
    3. University video core
5. Put the Linux image on a SDcard
6. Configure Linux as you see fit, but at least follow the steps on Linux on DE1-SoC

7. Get PuTTY and PSFTP installed on your Windows machine
   --or--
   SSH and SFTP on Mac OS X
8. I prefer to edit C in Notepad++ on the Windows machine, then PSFTP the code to the HPS SDcard directory to compile.
9. Edit Verilog/Qsys in QuartusII/Qsys and download to the FPGA using the Quartus loader
   (assuming the DE1-SoC config switch 0-5 is 010101)
   Use QuartusII v15.1
10. I suggest that you start with a running Quartus project based on the VGA game of life on the
    University Program DE1-SoC_Computer_15_1 page.

Assignment

- You will design the system in C on the HPS and Verilog on the FPGA. Don't use schematic entry or VHDL.
- The HPS will run the Conway CA state machine and control the USB mouse.
- The VGA resolution must be 640x480.
- When the HPS program starts, the screen will be blank (except for CA compute time), and the simulation stopped.
- SW0 on the FPGA will control the run/stop state.
- By clicking the mouse left-button (with SW1=0) you will draw/erase individual cells of the CA at the mouse location.
- By clicking the mouse left-button (with SW1=1) you will draw a PI form at the mouse location.
  The code below shows a PI drawn at the center of the screen.

```
// init a small pattern "PI"
   life[320][240] = 1;
   life[319][240] = 1;
   life[321][240] = 1;
   life[319][241] = 1;
   life[319][242] = 1;
   life[321][241] = 1;
   life[321][242] = 1;
```

- By clicking the mouse right-button you will draw a Gosper glider gun at the mouse location.
  Clearly you will need to do a bounds check first.
- All of the drawn cells should evolve according to Conway's game of life when the run/stop switch is in run state.

- The CA compute time for updating all the cells once should be displayed on the screen.
- There should be no flickering, tearing, or other video artifacts caused by the code running on the HPS.

Be prepared to demo your design to your TA in lab.

Your written lab report should include the sections mentioned in the policy page, and :

- A state-transition diagram for the main CA update state machine.
- A video of the VGA screen showing the evolution of an interesting CA configuration.
- Comment and speculate on Conway's game of life and Turing machines.
- A heavily commented listing of your C and Verilog design.

 February 2, 2017