**DISPLAY**

(Burhan and Yashika)

After designing a game from scratch, the most essential part left was to be able to display the results visually. Although there are many methods used in this era to display results, we are limited to the outputs of our FPGA board. Therefore, we only had two options to display the game on. The first one was using the seven segment display that is on the board of our FPGA. The second way was to use a side port that utilizes the Video Graphics Array (VGA) attached to the board. Using the VGA port was one of the goals that we had in mind, however, we realized the complexity of working with VGA connections and decided that using the Seven Segment display would be a better fit due to the time limitation we had for this project.

To get the display module working, multiple inputs were required in order for visual results to be presented. The inputs of this module will be the same outputs generated from the physics engine, map generator, and control panel. Therefore, the outputs from these modules must match the size of the inputs or data may be lost. A 16-bit input was used for the display as the map generator produced its final results in 16-bits.

As a start to display the inputs received, we will have different cases in our module that will describe various shapes of the current status of the game on each seven segment. Hence, a three bit decoder was used to indicate which case exists, which will then provide us with an 8-bit output to describe the shape. This is explained by activating and deactivating the specific segments of each display from A through G. The binary code will indicate which bit is on and which bit is off, using 0’s and 1’s, respectively.

The different cases of each possible output:

* **Unicorn down:** case when the unicorn is on the lower part of the display.
* **Unicorn up**: case when the unicorn is on the upper part of the display.
* **Obstacle down**: case when obstacle is on the lower part of the display.
* **Obstacle up:** case when obstacle is on the upper part of the display.
* **Unicorn up with obstacle down:** case when the unicorn is on the upper part of the display and the obstacle is on the lower part.
* **Unicorn down with obstacle up:** case when the unicorn is on the lower part of the display and the obstacle is on the upper part.
* **Dead:** case when the unicorn and the obstacle hit each other (Both up or both down on the same display)
* **Empty:** case when no obstacle is displayed.

**SHAPES ON THE DISPLAY**

char(0) \*\*EMPTY\*\*

char(12)\*\*UNICORN DOWN\*\*

char(66)\*\*UNICORN UP\*\*

char(99)\*\*OBSTACLE UP\*\*

char(92)\*\*OBSTACLE DOWN\*\*

char(94)\*\*OBSTACLE UP UNICORN DOWN\*\*

char(111)\*\*OBSTACLE DOWN UNICORN UP\*\*

char(118)\*\*GAME OVER\*\*



Display Unicorn down with a lower box obstacle

Since we are using eight seven segments, each segment must be used wisely since we do not have a lot of space. We decided to use the left most seven segment to hold the position of the unicorn. The rest of the segments will display the obstacles and rest of the cases. Since we have the most significant display for the unicorn mainly, we had to split the cases we had on two different decoders.

The First decoder is responsible for the following:

* Unicorn: This seven segment displays either the unicorn, unicorn with an obstacle, or the dead case.
* [Unicorn up](#7lzfs9nxcdqk)
* [Unicorn down](#bqnetboowy1n)
* [Unicorn up with obstacle down](#1hdistp8xgng)
* [Unicorn down with obstacle up](#isqy7maybpd)
* [Dead](#ubczvgm75n5p)

The input for this part is from the physics engine and the output is displayed at the MSB seven segment. The display works with the positive edge of the clock. We take the input to generate a three bit output for our first seven segment decoder. The three bit output decides what case is displayed on this seven segment.

The Second decoder is responsible for the following:

* Obstacles: The rest of the seven segments will either display the position of an obstacle or will display an empty case.
* [Obstacle up](#48cf63g205rz)
* [Obstacle down](#inqu6c6s91uw)
* [Empty](#b4k915bkgs3a)

The input for this part is also from the physics engine and the output is displayed at the rest of the seven segments. Just like the previous case, the display will also work with the positive edge of the clock. We take the input from the physics engine to generate a different three bit output which will be responsible for the rest of the seven segments. The three bit output will decide which one of the three cases shown above will be displayed.