Fishing Gold

Jixiong Deng

Xin Geng

Introduction:

**GAME INSTRUCTION**:

The game is controlled by switches and keys on DE1-SOC board. When the game is initialized, you need to press KEY[1] to continue the game panel. SW[9] and SW[8] control the movement of the hook. Pressing KEY[3] can drop the hook. When the game is done, pressing KEY[1] and KEY[2] can replay the game.

**MOTIVATION OF THE GAME**:

The idea of this project comes from a website game, “Golden Miner”. Basically, in that game, player controls a hook which is located at the center of the top of the screen and always swinging from 0 degree to 180 degrees. The hook will go down along with the direction at the instant when player presses the drop button, and will return when it reaches its maximum length or a stone. There are many kinds of stone, like rock, cobble, gold and diamond, on the screen. Each of them represent different score value. The goal of that game is to get a target score in a limited time. If player does get that score, he or she will lose the qualification to the next mission.

Our game, “Fishing Gold”, inherits most of factors of “Golden Miner”. To make the game easier to control, we modify the hook part. Now, it is movable from the left side of the screen to the right side. When player decides the position and presses the button, the hook will drop vertically until it catch a gold or the bottom of the screen. And then the hook will appear at the initial position again. We simplify the types of stone. When the game gets started, there will be 5 golds generated randomly on the screen. When player gets any one of the gold, a new gold will be refreshed randomly. Every gold here in this game worth 30 points and the score will be recorded on the DE1-SOC board. Player aims to get 150 points to win the game and watch the congratulation sentence. Player can press the start key to play a new game.

The design:

**BLOCK DIAGRAM:**

True

New game signal

**False**

Start signal

**False**

**True**

**True**

**False**

Score = 150

Move signal = 1

**True False**

**False**

Collect gold

**True**

**FUNCTION OF PARTS:**

Background:

When press the KEY[2] to start the game, the screen will transfer from the beginning graph to the background graph. In this case, a memory block is create to store a 160\*120 pixels’ graph as the game background. X-axis and Y-axis of the screen are assigned as the address of the memory block. Each address has a unique color output. A counter counts from 0 to 19119 to record this process as a feedback to the FSM to determine whether go to next state or not.

Gold:

Gold part is next to the background part. There are 5 “random counters” to generate random position of golds. The idea is to make 5 independent counters always make addition and return to 0 when the number become to a specified value. The initial and final value of each counters are different to make sure they have different values at the same time. These counters begin to work as long as the start of the program. There are 5 register to record the random position when the game starts or a gold is be caught. A memory block is also necessary in this part. Instead of using X-axis and Y-axis, we use a counter as the input of memory block. The address will be the sum of gold position and counter’s value.

Hook:

Hook part consists of serval subparts. When the game gets started, a hook will be drawn at the initial position. Then, the hook goes into an idle state until you give it a signal to right or to left. When the signal is given, a state works for erasing previous hook will be activated. The principle is same as drawing the hook, but the color is always black. After that part, the FSM shall go to part to calculate new position. The Y-position of the hook is fixed at this time, “left” signal will enable the state to do subtraction on X-position. Inversely, the “right” signal will enable the state to do addition on X-position. Every time, the part will use a memory block which stores the graph of hook and same method as gold part to get the color information of each position. Besides that, there is a “delay” counter to make program idle until the counter reaches a value. This design controls the speed of hook. When the “drop” signal is given, the X-position is no longer changed, and Y-position is always do addition before it reaches its maximum length or a gold. The difference between left and right control, the previous picture won’t be erase, as the dropping rope. If it reaches its maximum length without touching any gold, the hook will bouncing back which the Y-position will always do subtraction until the original position.

Draw:

All of background part, gold part and hook part pass their position information and color information into the draw part. The draw is controlled by FSM to decide which graph to draw because VGA module only can draw one pixel at every clock edge.

Touch:

Gold part and hook part pass their position information into the touch part. The touch part compare with both position to see whether they collide or not. If a hook touches one of golds, a touch signal will pass to the FSM and reload a new random gold position and redraw the background, golds and hook. The touch signal also will be received by score part to record the score.

Score:

Score part only receive the signal from touch part. Every time, the touch signal is activated. The counter in the score part will do addition to get a new score.

Work Demonstration:

Self-examination:

Underestimate of the Workload:

Poor Corporation: