

## CS M152A Project Proposal

### Gameplay

For this project, we are aiming to create a slot machine, where the player has the potential to multiply their bets by 10 if they get all 7's, multiply their bets by 3 if they get any other set of 3 matching numbers, and lose their bets otherwise.

Our gameplay will follow this sequential process -

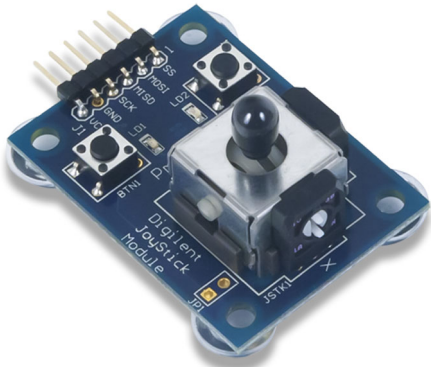

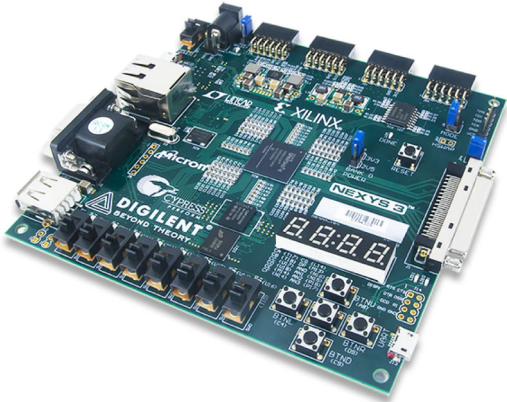
1. When the player first starts the game, segment display shows the default amount of money that person has (i.e. \$100).
2. The player then inputs the amount of money they want to bet using the number pad (this amount has to be less than or equal to the amount of money they currently have).
  - a. If the user inputs more money than they can bet, output "E's" to denote an error for five seconds. Afterwards, the user has to input a valid bet.
  - b. Otherwise, take in the amount of money the user inputted to bet.
3. Afterwards, set the seven segment display to all 0's. This will denote that the slot machine is ready to begin. The user then has to engage the lever to have the random numbers appear.
4. After 5 seconds, have the randomness stop.
  - a. All 7's means that the player multiplies their betting amount by 10.
  - b. Another triple means that the player multiplies their betting amount by 3.
  - c. Anything else, lose your initial bet
  - d. Display this for 5 seconds
5. If you win, flash with the amount of money you gained.
6. If you lose, show minus sign with amount of money you bet/lost
7. After 5 seconds of above, the player is prompted back to how much money they have after this round and the gameplay repeats.

---

### Features and Technical Components

Our group is aiming to implement a wide variety of technical modules and features to make our slot machine as practical and realistic as possible. The table below demonstrates the various technical modules and features we are planning to implement:

<b>Pmod JSTIK Digilent Joystick Module</b>	We are planning to use the Digilent Joystick Module to simulate a slot machine "pull". In a technical aspect, we are going to provide hardware randomness everytime the FPGA board receives a signal that the joystick has been moved. We are not necessarily going to care about what direction
--	--

 <p>A blue printed circuit board (PCB) labeled "Digilent JoyStick Module". It features a black joystick with a silver base and a black knob. There are several push buttons and a potentiometer on the board. A 5-pin header is visible on the left side.</p>	<p>the joystick moves at, but just consider an action as a pull whenever the joystick moves at any direction.</p>
<p><b>Pmod KYPD 16-button Keypad</b></p>  <p>A blue PCB labeled "Pmod KYPD" featuring a 4x4 grid of 16 black buttons. The buttons are labeled with numbers 1-16 and letters A-D. It has a 5-pin header on the left and a 16-pin Pmod header on the right.</p>	<p>We will be using the 16-button Keypad in order to allow the player to bet in the slot machine game. We will be using the "E" key as a "Confirm/Enter" key whenever the player(s) wants to confirm their bet and use the "C" key as a "Clear" key whenever the player(s) want to clear their bet amount.</p>
<p><b>Nexys 3 Spartan-6 FPGA Trainer Board (Seven-Segment Display)</b></p>  <p>A green PCB labeled "NEXYS3" and "XILINX". It features a large black integrated circuit (FPGA) in the center. There is a seven-segment display showing "8.8.8.8". Various other components like capacitors, resistors, and connectors are visible on the board.</p>	<p>We will be using the seven-segment display embedded within the Nexys FPGA board to showcase how much money the player has, how much money the player betted, the slot machine pull values, and how much money the player won/lost. The display will blink depending on what mode the player is on. If the player is betting or wins any money, the display will blink the numbers; otherwise, the display will remain in normal mode.</p>
<p><b>Hardware Randomness</b></p>	<p>We are planning to simulate hardware randomness</p>

Members: Rahul Mallick

Eric Choi

Sahiti Gabrani

	by extracting a portion of the master clock. Since there is no way to simulate "true" hardware randomness, we are simply going to randomize to the best of our capabilities by using the clock count value(s).
--	--

Assuming that we have completed all our original goals, our **stretch goal** is to implement another technical module and feature. The technical module and feature is listed in the table below:

<b>Nexys Spartan Board (LED Lights)</b>	If we finish with all the goals listed above, we are planning to incorporate the LED lights in the Nexys Spartan Board in order to "flash" whenever the player wins.
---	--

We believe that these technical modules are challenging enough to allow us to showcase our creativity.

---

## Timeline and Correctness Rubric

We have created a timeline for our project based on the date for the final demo with the TA and the complexity of each component.

Week	Date	Task to be completed
Week 8	05/22/2023	Complete Project Proposal for Slot Machine
	05/24/2023	Receive a signal from the joystick
Week 9	05/29/2023	Get values from the number pad
	05/31/2023	Display random number on the seven segment display after receiving acceptable inputs from joystick and number pad
Week 10	06/05/2023	Logic for winning or losing and fix any bugs (if time permits: LED lights blinking when player wins)
	06/07/2023	Work on the report and Demo the project!

Based on the timeline and the difficulty of working on each component involved, we have created a correctness rubric for grading purposes. The correctness rubric breaks up the project's demonstration into four parts and is given a percentage of the grade in accordance with the task's complexity.

Members: Rahul Mallick

Eric Choi

Sahiti Gabrani

Component Implemented	Percentage of Grade
Ability to use joystick to engage with slot machine	25%
Successful input from the number pad	25%
Proper implementation of the seven-segment display (no inverted numbers or incorrect segments)	20%
Game mechanics (logical flow from one state to the next without any hiccups) and determining whether player lost or won	30%

<https://github.com/jcherianucla/UCLA-CS-M152A/tree/master/Project>

