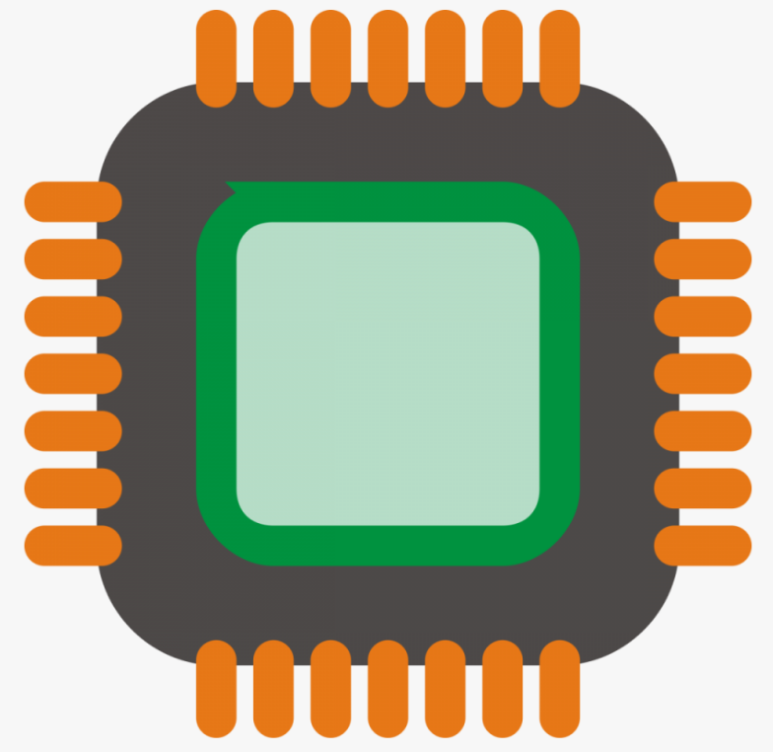
CS210 MINI

PROJECT – II



GROUP MEMBERS-

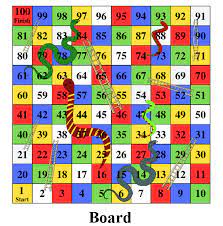
VARDHAN GACCHE (2101CS80)

SAIRAM PAILA (2101CS70)

VIKASH KUMAR VERMA (2101CS82)

About the project

In this project we have made a snake and ladder game using the dE1-SoC board. The code written is compatible with ARM A9 Cortex processor. We run the program in the Intel FPGA Program. The player will be able to play this game in the terminal of the Intel FPGA Program using buttons in dE-1 SoC to roll the die.



THE GAME

This game is a Computer VS You game. When the program is run on the Intel FPGA Program, a welcome message is displayed and the user has to enter his name and the game will start. Now a board will appear which will be the map of the snake and ladder game. In this map the current position of the user is displayed by ‘O’ and current position of computer is displayed by ‘X’. Below this grid the current position of both the user and computer is displayed. During each round of game play, the player goes first and then it’s the computer’s turn. During the player’s turn, the player will roll the dice by pushing 1 of the 4 push buttons on the DE1-SoC board. The game will print the result of the dice roll and the destination position after each turn. The game will also print the resulting board after each turn.Like in the actual game of snake and ladders there are snakes on some positions, if current position is on the head of a snake then the user position will now be the tail of the snake and also a message will be displayed saying that you have steeped on a snake.

The Snakes and Ladders game is driven by 2 players advancing throughout the board by rolling a dice. Each player could potentially land on a space that is either a snake or a ladder. In our game, one of the players is the computer. The Snakes are labelled with S meaning when the current position of any of the user or computer is in the cell having S it means that it has stepped on the snake and now the new position will be the tail of the snake. The Ladder spaces, labelled with L.. The last space which is denoted by $$ is the winning space.

CODE & INTERACTION

For this project, we generated a static board, meaning every run of the game will result in the exact same board where the ladders and snakes are at the same positions as before. This is demonstrated in all the screenshots in the Results section. Some of the screenshots were not taken within the game, but all the boards are exactly the same. This is the result of hard-coding the exact destination position of every snake and ladder space. In the nextMove() function in move.c, we check the given original position and returns the destination position if the space is a snake or ladder, if not then the function just returns the same position.

The game board generation could potentially be randomized to make the board different every game. This would make some parts of the game more exciting for the user. However, for the resulting board to be fun, there are many conditions that needs to be satisfied such as: there should be ladders and snakes of varying length, there should be more snakes towards the end of the board and more ladders towards the beginning of the board, and there should be a minimum and maximum length for each snake and ladder. These conditions are much harder to satisfy if the board is truly randomized, we could potentially generate a board where all 6 of the spaces prior to the winning space are all snakes which would result in a never-ending game. Using a static board guarantees that all the snakes and ladders are strategically placed throughout the board.

We designed this game by separating the different functionalities into different files and helper functions to better organize and assign responsibilities. The main program simply calls the helper functions to advance the game from one step to the next. We split the project into the following categories and functionalities:

* art: display winning ASCII word art for the player or the computer
* board: print the board based on the player’s and the computer’s position
* timer interrupts: configurations for the timers, interrupt service routines, and exceptions
* main: dice roll, control logic of player and computer turns
* move: determine destination position on a snake or ladder space

The snake and ladder game has many possible sources of errors. There could be an off-by-one error with rolling the die where the resulted value ranges 0 to 5 instead of 1 to 6. The resulting board after a turn does not display the player or the computer advancing through the board with its corresponding dice roll. Many more errors could happen during the implementation of this game. By organizing this project’s code into smaller identifiable categories, it is much easier to debug and investigate the source of the errors. If the game never produces a dice roll of 6, the error must be within the rollDice() function within the main file. If the displayed board is incorrect, then the error must be within the board file.

INDIVIDUAL CONTRIBUTION

*Sairam Paila(2101CS70)*

I helped the team to design the board and overall moves in the Snakes and ladders game. I also helped them to edit the video and contributed equal support to other members’ work too.

*Vardhan Gacche(2101CS80)*

I helped the team to design functions related to rolling the die and other moves and helped them to establish well interaction between hardware and software. I also helped them with report writing and contributed equal support to other members’ work too.

*Vikash Kumar Verma(2101CS82)*

I helped the team in writing codes for all the moves in the game with deciding moves and positions for snakes and ladders. I also helped them in ppt making and contributed equal support in other members’ work too.