CPSC 359 – Fall 2014
Assignment 3: Raspberry Pi Video Game

based on J. Kawash - 2014

due: 0900h 24-Nov-2014

Background

This assignment will expand your skill set with the Raspberry Pi. It will require that you use the I/O capabilities of the Pi to get button states from and SNES game controller. You will work with a video display controlled by the Pi. Finally, you can use the power of a (slightly) higher level language to develop code that works directly with I/O devices. Your TAs will help you with the required knowledge for the I/O devices.

Objective

Implement a video game (*in C*) that requires the player to travel through a maze with the goal of reaching the exit door within a specified number of moves. Solving the maze will require the player to move through the maze, collect keys, open doors that block the path, and open the exit door before the actions remaining counter runs out.

Game Logic

The game environment is a finite 2D n x n grid (see Figure 1).

- Each grid cell contains either a wall tile or a floor tile
  - Floor tiles may be marked as player start or key
  - Wall tiles may be marked as door or exit door
- A game map is an instance of the game environment (for a value of $n > 16$)
  - Specifies the maximum number of player actions for the map
  - Each cell of the grid is either a wall or floor tile
  - The cells on the edge of the map are filled with wall tiles
  - One floor tile is marked player start and one wall tile is marked exit door
  - There are k floors marked as key and d wall tiles marked as door (1 ¡ k ¡ d) o No floor tile is adjacent (in the cardinal directions) to more than one door tile
- A game state is a representation of the game as it is being played, and contains:
  - An instance of a game map
  - The players position on the game map
    - Initialized to the position of the floor tile marked player start
  - The number of actions remaining that the player can perform
    - Initialized to the game maps maximum player actions
  - The number of keys collected by the player (initialized to zero).

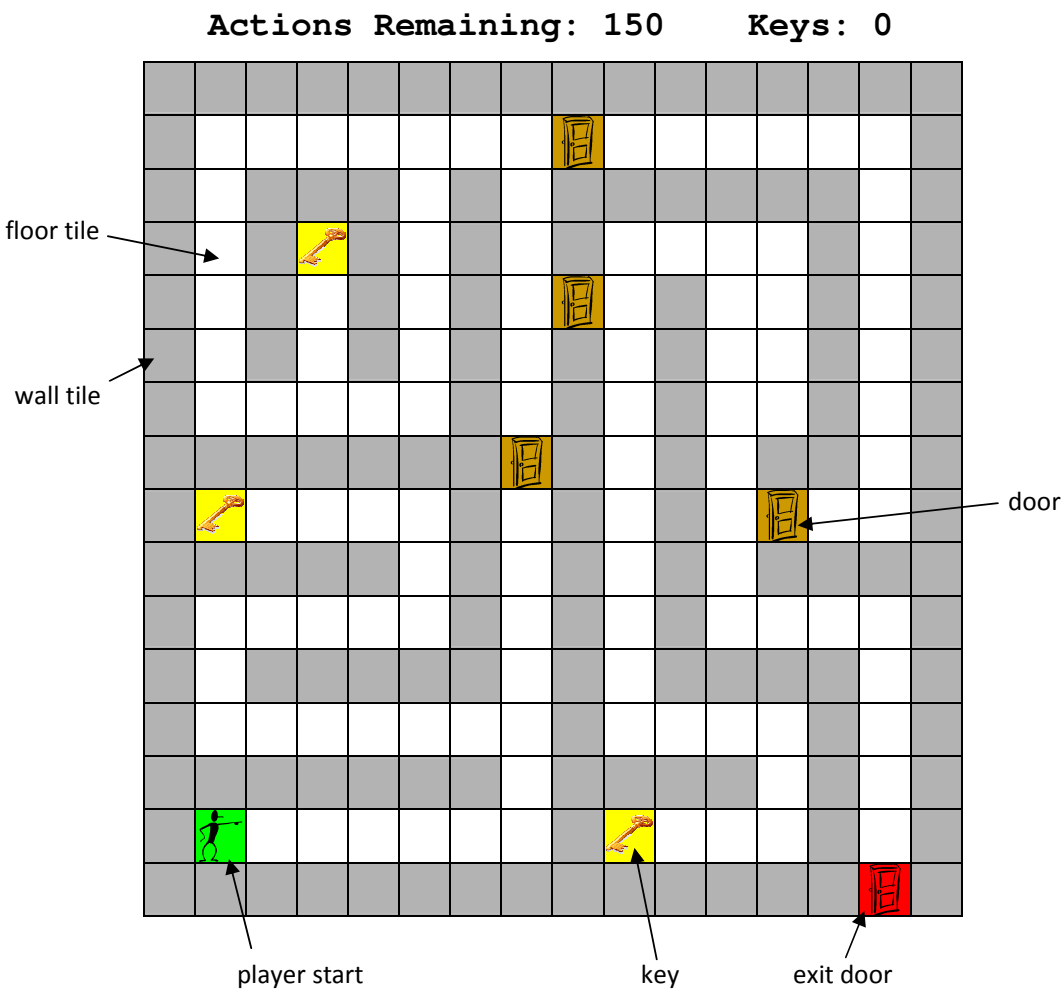

Figure 1: Example game rendering.

  - A win condition flag (set if and only if the player opens the exit door)
  - A lose condition flag (set if and only if actions remaining equals zero)

The game transitions into a new state by the player performing an action

- Only transition to a new state if the player performs a valid action
  - Decrement by one the actions remaining counter in the new game state
- Action: Move by one cell in one of the four cardinal directions (up, down, left, right)
  - Move action is valid only if the destination cell contains a floor tile
  - Results in the players position being set to the position of the destination cell
  - Moving into a floor tile marked as key will result in:
    * The number of keys collected being incremented by one
    * The removal of the key marking on the destination floor tile
- Action: Open a door or the exit door with a key

  - An open door action is valid only if:
    * Player is adjacent to a wall tile marked door or exit door
    * Number of keys collected is greater than zero
  - Opening a door results in:
    * Adjacent wall tile marked as door or exit door is replaced by a floor tile
    * Number of keys collected is decremented by one

Game Interface

Main Menu Screen

- The Main Menu interface is drawn to the screen
  - Game title is drawn somewhere on the screen
  - Creator name(s) drawn somewhere on the screen
  - Menu options labeled Start Game and Quit Game
  - A visual indicator of which option is currently selected
- The player uses the SNES controller to interact with the menu
  - Select between options using Up and Down on the D-Pad
  - Activate a menu item using the A button
  - Activating Start Game will transition to the Game Screen
  - Activating Quit Game will clear the screen and exit the game

Game Screen

- The current game state is drawn to the screen
  - Represented as a 2D grid of cells
    * All cells in the current game state are drawn to the screen
    * Each cell is square (width = height), and at least 16x16 pixels
    * 2D grid should be (roughly) in the center of the screen
  - Each different tile type is drawn with a different visual representation
    * ie: wall, floor, door, key, exit door, player start and player
    * Minimally, cells are filled with different colours based on tile type
  - Actions Remaining and Keys Collected are drawn on the screen
    * A label followed by the decimal value for each field
  - If the Win Condition flag is set, display a Game Won message
    * If the Lose Condition flag is set, display a Game Lost message
    * Both messages should be prominent (i.e: large, middle of screen)
- The player uses the SNES controller to interact with the game
  - Pressing up, down, left or right on the D-Pad will attempt a move action o Pressing the A button will attempt to perform an door open action
  - All buttons need to be released before a new action can be performed o Performing a valid action will require the game state to be redrawn
  - Pressing the Start button will open a Game Menu

	Main Menu Screen	
1a	Draw game title and creator names	2
1b	Draw menu options and option selector	2
1c	Select between menu options using up/down on D-Pad	1
1d	Press A button with Start Game selected to start game	1
1e	Press A button with Quit Game selected to exit game	1
	Game Screen	
2a	Draw current game state	
	All cells drawn according to interface specifications	5
	Actions Remaining and Keys Collected drawn	3
	Game Won message drawn on win condition	1
	Game Lost message drawn on lose condition	1
2b	Draw game menu	
	Filled box with border in center of screen	1
	Draw menu options and option selector	2
	Erase game menu from screen when closed	2
2c	Interact with game	
	Use D-Pad to move player (if move action is valid)	4
	Press A button to open doors (if open action is valid)	3
	Press Start button to open game menu	1
	Press any button to return to main menu (game over)	1
2d	Interact with game menu	
	Use up / down on D-Pad to change menu selection	1
	Press A button on Restart Game; resets the game	1
	Press A button on Quit; returns to main menu	1
	Press Start button to close game menu	1
	Well structured code	
3a	Use of functions to generalize repeated procedures	5
3b	Use of data structures to represent gamestate,etc.	5
4	*Well documented code*	5
	Total	50pts

Table 1: Grading rubric for assignment 3.

    * Two menu items: Restart Game and Quit
    * Visually display menu option labels and a selector
    * Menu drawn on a filled box with a border in the center of the screen
    * Normal game controls not processed when Game Menu is open
    * Pressing Start button will close the Game Menu
    * Press up and down on D-Pad to select between menu options
    * Pressing the A button activates a menu option
    * Activating Restart Game will reset the game to its original state
    * Activating Quit will transition to the Main Menu screen
- If the win condition or lose condition flags are set
    * Pressing any button will return to the Main Menu

Evaluation

See the grading rubric in Table 1.

You may work individually, or with a partner, but no larger groups.

Programs that do not compile cannot receive more than 5 points. Programs that compile, but do not implement any of the functionality described above can receive a maximum of 7 points.

Submit a .tar.gz file of your entire project directory (including source code, make file, build objects, kernel.img, etc) to your TA via the appropriate dropbox on Desire2Learn. If you are working with group members in different tutorial sections, choose just one TA to submit to. You will also need to be available to demonstrate your assignment during the tutorial of the TA you submitted the assignment to during the week of November 24.

Late work

After the deadline and up to 24hrs late: -10pts. After 24hrs and up to 48hrs late: -25pts. Over 48hrs late: -50pts, i.e., no assignment will be accepted beyond 48hrs after the deadline.

Plagiarism

Work must me the sole work of the individual(s) submitting the work.

You may find that the task for this assignment is similar to work found elsewhere. Nevertheless, you should go through the process yourself, and submit your own work. If you do borrow from other sources, cite the source and clearly indicate what you have borrowed, keeping in mind the design must be substantially your own. If you cite your sources, worst case you may receive a reduced grade for borrowing too much. If you borrow, but do not cite, that is plagiarism and academic misconduct, and carries severe penalties as determined by the Faculty of Science.

As a guideline, consider the 20-minute rule. Talk with your colleagues and consult other sources (cite them please). Wait at least 20 minutes, then do your work to be sure that it is your own. Less then 20 minutes usually means that you are merely copying work from the original source.