CPUlator Sokoban User Guide

Taym Alshoghri

Contents

Section 1: Introduction

1.1 - Introduction to Sokoban

Sokoban is a puzzle video game in which the player must push boxes in a grid onto a target. Successfully pushing each box onto a target completes the puzzle and allows the user to move on to a new, predetermined grid. The game's difficulty stems from the fact that the only interaction a player can have with a box is to push it. This means that accidentally pushing a box into a wall could mandate a restart from the user since there would be no way to pull it back.

1.2 - Brief Comparison With CPUlater Sokoban

In CPUlater Sokoban, all of Sokoban's push mechanics are present, and the player still has to move each box onto a target to win. However, the levels are not predetermined, and each run of the program will generate a completely unique set of box/target positions. The size of the grid can be modified by the user, and the number of boxes increases as the grid size increases. CPUlater Sokoban also boasts a competitive local multiplayer mode where an unlimited number of players can take turns competing for the smallest amount of actions/moves to solve a level.

Section 2: Playing The Game

2.1 - Setting up CPUlater

Before the following steps, ensure that a .s file of the game has been downloaded. The website used to run the game is called CPUlater, and can be found by clicking on the following link; (https://cpulator.01xz.net/?sys=rv32-spim). Once CPUlater is open, hover your mouse over the 'File' dropdown button near the top of the screen [Figure 1]. Then, click 'Open…' [Figure 1] and find the location of the .s file of the game that was downloaded. Finally, double click the .s file to load the necessary data into CPUlater.

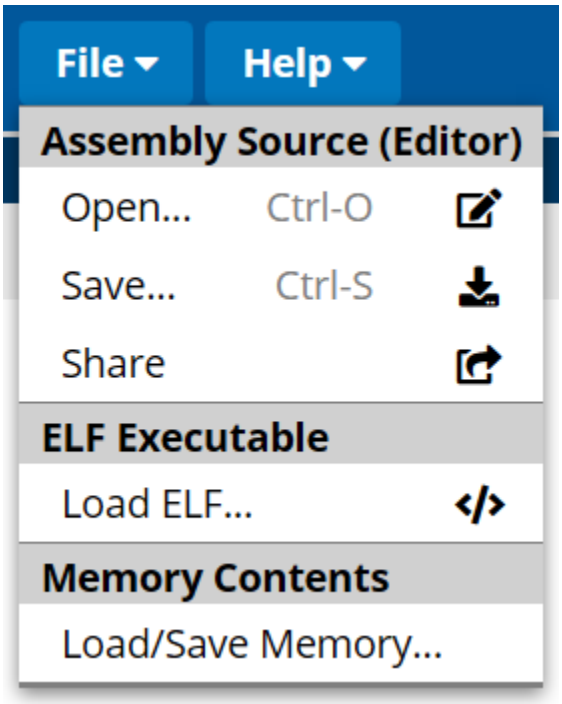

Figure 1: The 'File' dropdown menu containing the 'Open…' button.

Once the data has been loaded, click the 'Compile and Load (F5)' button in the 'Editor (Ctrl-E)' section [Figure 2]. You can find this section underneath the top bar where 'File' [Figure 1] was located. Alternatively, you could simply press the 'F5' key on your keyboard.

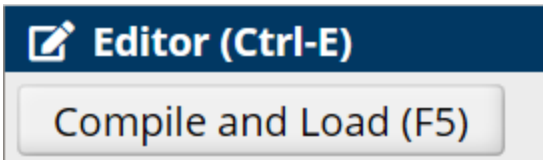

Figure 2: The ‘Compile and Load (F5)’ button.

After clicking the ‘Compile and Load (F5)’ button or pressing ‘F5’, direct your attention again to the blue bar near the top of the screen. Here, there are three buttons of significance [Figure 3];

1. ‘Continue’ will start the game when clicked, or continue the game if ‘Stop’ was clicked.
2. ‘Stop’ will pause the game when clicked.
3. ‘Restart’ will restart the game when clicked, creating a new board in the process. Note that ‘Continue’ will need to be clicked again after ‘Restart’ is clicked.

Figure 3: The ‘Continue’, ‘Stop’, and ‘Restart’ buttons.

To begin playing, move your mouse over the ‘Continue’ button [Figure 3] and click it.

2.2 - Understanding Console Outputs / Inputs

The console (sometimes called the terminal) is where the visual part of the game will be displayed. The console can be found at the very right of your screen, under the ‘Devices’ and ‘Terminal’ headers [Figure 4]

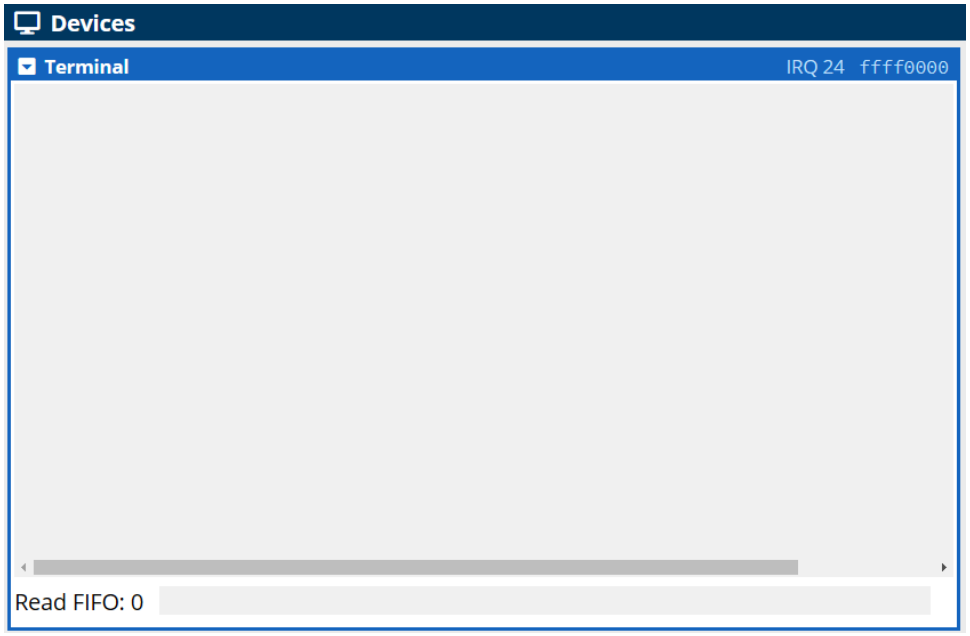

Figure 4: The console

To input characters or numbers into the console, click on the console and then type the desired input key on your keyboard.

Now that we know how to use the console, we can explain the meaning of the various images and text you will see when playing the game.

2.2.1 - Player Amount Prompt

Right after starting the game, you will be prompted for the amount of players that will be playing the game. As mentioned in 1.2, each player will have a turn solving the same puzzle and the player that does it in the least amount of moves wins. After receiving the prompt, type the integer number of players you are playing with and then click ‘Enter’ on your keyboard. If

you want to play alone, simply type ‘1’. Any invalid input, such as a string or negative number, will cause the game to be run with only one player.

While there is no hard limit on the amount of players that can play, the user should consider that entering an absurdly large number could cause uncontrollable errors in the game.

2.2.2 - Decoding the Board Characters

Once a player amount has been entered, a grid will be displayed in the console [Figure 5]. The displayed letters mean the following;

1. [X] Represents the character the user controls.
2. [O] Represents a wall that the player is not allowed to move into.
3. [#] Represents a box that the player can push.
4. [*] Represents a target that the player can walk on top of and push boxes onto.

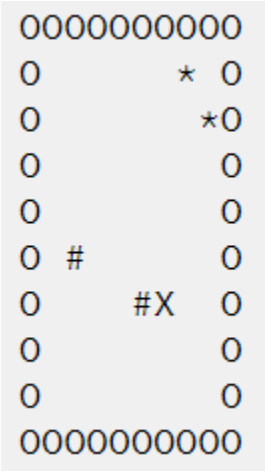

Figure 5: An example of a generated grid

2.2.3 - Player Controls

The player can input the following letters into the console while the game is running;

1. [w] to move the character upwards
2. [a] to move the character to the left
3. [s] to move the character downwards

4. [d] to move the character to the right
5. [r] to reset the board to its original state

The player can move onto any empty space and target, but cannot move on top of a wall. If the player moves on top of a box, the box will be pushed in the direction the player is moving. However, the game does not allow a player to move on top of a box if there is no space for the box to go, i.e. the box is against a wall or another box. If the player tries to move into a wall or push a box that has no space to move, an 'Invalid Move' text will appear. Otherwise, if the movement was successful, the current player will count as having taken a move. Note that resetting the board also counts as a move for the current player.

Any invalid input (a letter that isn't 'wasd' or 'r') or invalid move will show an 'Invalid Input' or 'Invalid Move' error, but not count as a move for the current player.

2.2.4 - Player Victory

To win the game, the current player must push each box so that it is on top of a target. Once each target has a box on top of it, a text will appear stating your player number and the number of moves it took for you to win [Figure 6]. If there are multiple players, the board will then reset to allow the next player to complete the puzzle. If all players have finished the puzzle or if there was only one player playing, the scoreboard will appear.

In the case that a player has accidentally rendered the puzzle unsolvable, e.g. a box is pushed into a corner, that player can enter 'r' to try again. This will reset the board but maintain the number of moves that player has previously done.

```
PLAYER 1 WON IN 30 MOVES
```

Figure 6: An example message displayed when a player wins.

2.2.5 - Scoreboard

Once all players have won the game, a scoreboard will be displayed ranking each player by their number of moves [Figure 7]. The player at the top is the one with the least moves, and thus, the winner. Each player is numbered by the order that they played in, e.g. the user who finished the puzzle second is ‘PLAYER 2’

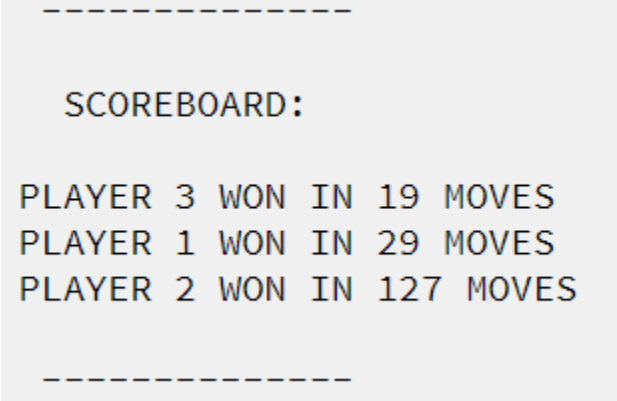

Figure 7: An example scoreboard displayed when every player has won

2.2.6 - Playing Again

After the scoreboard is printed, the game will ask the user if they would like to play again. If the letter ‘r’ is typed in the console the game will generate a new board and ask for the number of players again. If anything else is inputted, the game will end. If you would like to play again after the game ends, press the ‘Restart’ and then ‘Continue’ button as outlined in section 2.1.

Section 3: Customization Options

To customize the game, direct your attention to the text under the ‘Editor (Ctrl-E)’ section [Figure 8]. Hover your mouse over the text and scroll down until you see the modification you would like to make. Then, click on and replace the text as required. Examples of modifications can be found in the section 3 subsections.

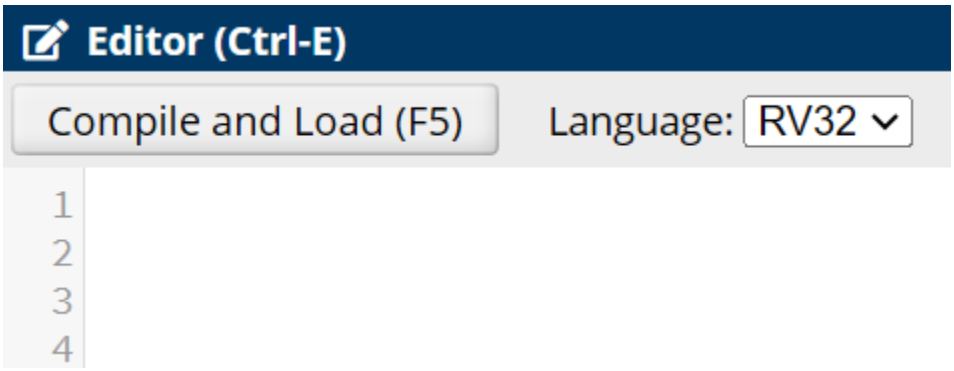

Figure 8: The location of the Editor’s contents. Note that the lines 1, 2, 3, and 4 normally contain text.

3.1 - Changing the Grid Size

To change the size of the grid, find the line that starts with ‘gridsize: .byte’ [Figure 9]. To the right of .byte, you will see two numbers. The first number represents the grid’s width and the second number represents the grid’s height. Either of these values can be changed to modify the grid size as desired, but to maintain functionality, the values should be numbers greater than three. While there is no hard limit on the size of the grid, the user should consider that entering an absurdly large number could cause uncontrollable errors in the game.

Figure 9: The location of the ‘gridsize’ component.

3.2 - Modifying the Grid Size to Box Number Ratio

As mentioned in 1.2, the number of box/target pairs scales with the width and the height of the board. To change the ratio that determines the number of box/target pairs, find the line that says ‘numBoxes:’ [Figure 10]. Two lines under ‘numBoxes:’ will be ‘li t0, 3’. Increasing the ‘3’ will decrease the amount of box/target pairs that spawn and decreasing the ‘3’ will increase the amount of box/target pairs that spawn. The ‘3’ should not be replaced with anything but a non-zero positive integer.

```
438 # Arguments: None
439 # Return: The number of boxes for this board based on gridsize
440 numBoxes:
441     # numBoxes = gridsize // t0
442     li t0, 3
```

Figure 10: The location of the box/target to grid size ratio