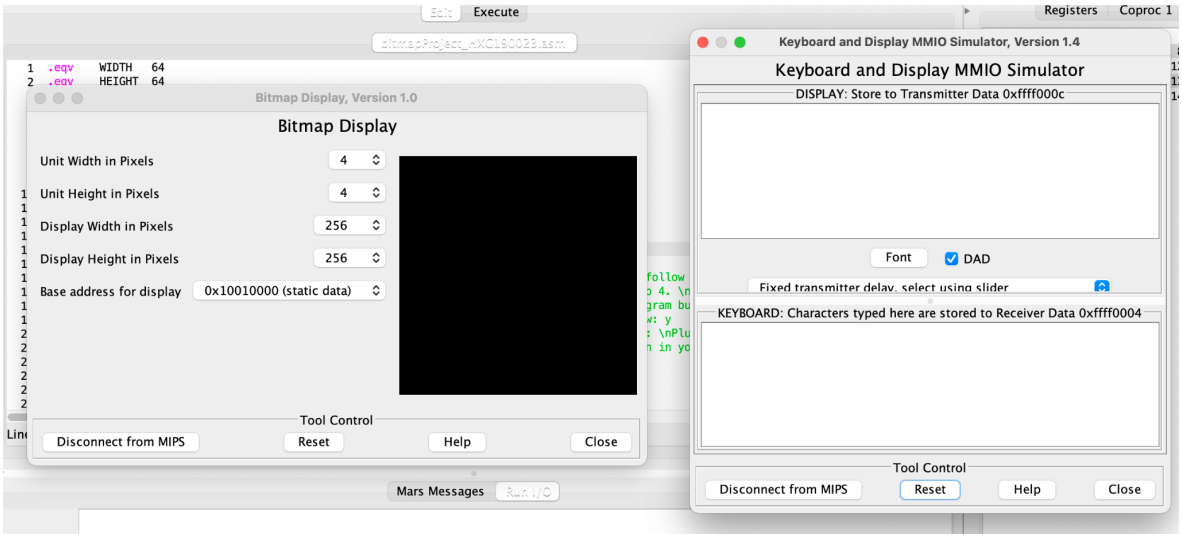
Bitmap Project Report

Name: Harsh Gopalan | Class: CS2340. 003

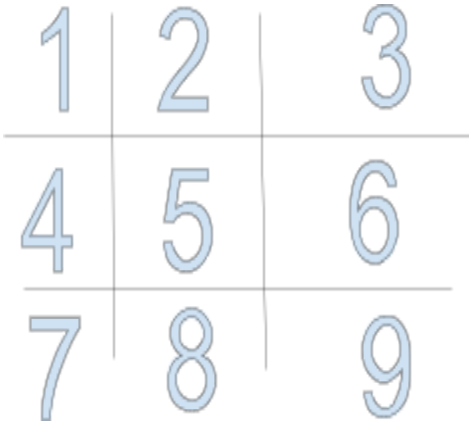
Instructions:

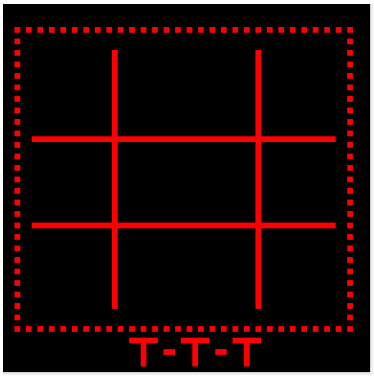
Please make sure to follow the instructions carefully.
Please read all the instructions first and watch the video run-through, then begin playing the game.
Watch the video run-through of the game here: https://youtu.be/OkA6gC8roo8\

1. Set both the Unit Width and Unit Height in Pixels to 4.
2. Set both the Display Width and Display Height to 256.
3. Set the Base address for display as static data.
4. Make sure to connect the Bitmap Display and the Keyboard & Display MMIO Simulator to MIPS.



5. Once the program has been assembled and the run program button is clicked, the gameboard will be drawn.
6. Please wait till the entire gameboard is drawn until you begin playing the game.
7. Once the game board is fully drawn, the first player can start by picking a box to draw their shape.
8. The boxes are labeled 1-9, horizontally. (1-3) first row, (4-6) second row, (7-9), third row.
9. Make sure to select the box number first.





Bitmap Project Report

Name: Harsh Gopalan | Class: CS2340. 003

10. Once the box number is selected, do not click enter or any other key as this will mess up the program.
11. Continue reading the instructions once the box number is selected.
12. Next, select one of the colors listed below by clicking the correct key on the keyboard:

Red: r Blue: b Magenta: m White: w Yellow: y
Cyan: c Green: g Orange: o Purple: u Gray: i Maroon: n

```
beq     $s1, 110, colorMaroon # input n
beq     $s1, 105, colorGray # input i
beq     $s1, 117, colorPurple # input u
beq     $s1, 111, colorOrange # input o
beq     $s1, 114, colorRed # input r
beq     $s1, 109, colorMagenta # input m
beq     $s1, 103, colorGreen # input g
beq     $s1, 98, colorBlue # input b
beq     $s1, 119, colorWhite # input w
beq     $s1, 121, colorYellow # input y
beq     $s1, 99, colorCyan # input c
```

13. Once again continue reading the instructions once the color is selected and do not press enter or any other key until you read the next set of instructions.
14. Finally, select one of the five shapes listed below:

Plus Sign: p Minus Sign: s Square Box: o X-Shape: x
Smile-Face Shape: f

```
beq     $s1, 112, settingPlus # input p  # drawPlusShape  # Will draw a plus sign shape
beq     $s1, 115, settingMinus # input s   # Will draw a minus sign shape
beq     $s1, 113, drawBoxShape # input q
beq     $s1, 120, drawXShapeLeftDiagonal # input x
beq     $s1, 102, setSmileyFaceTop # input f
```

15. Once this is done, your selected shape will be drawn in your chosen color at your chosen box.
16. When the shape is drawn, a sound effect can be heard, so make sure to turn up the volume on the device that will be running the program

Bitmap Project Report

Name: Harsh Gopalan | Class: CS2340. 003

17. Make sure to do the steps in that order, as the program will not work if it is not done in that order.
18. Once the first player has finished their turn, the next can go, and the game can be played until the game ends.
19. The program will not let the user know who won, but looking at the game-board, this can be figured out.
20. If the entire gameboard gets filled up, the program will automatically exit, given that the users did not have any mistakes while playing.
21. To exit the program at any point, press the spacebar.

Overview of the Program:

General Description of Program:

This program is a simulation of a tic-tac-toe game. It allows the users to select a shape, the shape's color, and the box in which it can be drawn. The program will also let the user know as they run the program, which players should be playing, and what action they should be performing. For example, the program will let the user know that it is Player 1's turn and that they need to now select a color.

The program can be played until one of the users has won the game. Since this is a rudimentary simulation of tic-tac-toe, this program does not let the user know who has won the game, but this can be easily figured out by looking at the gameboard.

Logic/Implementation of Program:

This program makes use of various function calls, loops, branches, to flow back and forth, by taking user input to do the correct actions and print the correct output on the bitmap display. The program has an "end" point when the gameboard gets filled, up and this results in the exit syscall being called. The specifics of the program including the pseudo-code will be explained later in the report.

Pseudo-Code & Flowchart of Program:

Pseudo-Code:

- The instructions will be printed to the console
- Each side of the game board is drawn
- The border animation is drawn
- The logo is drawn at the bottom of the game board

The above pseudocode instruction uses a series of loops to print the borders and the game board in an efficient manner, but as a whole, the program, up to this point, will be small functions that branch to the next once the loop is over.

Bitmap Project Report

Name: Harsh Gopalan | Class: CS2340. 003

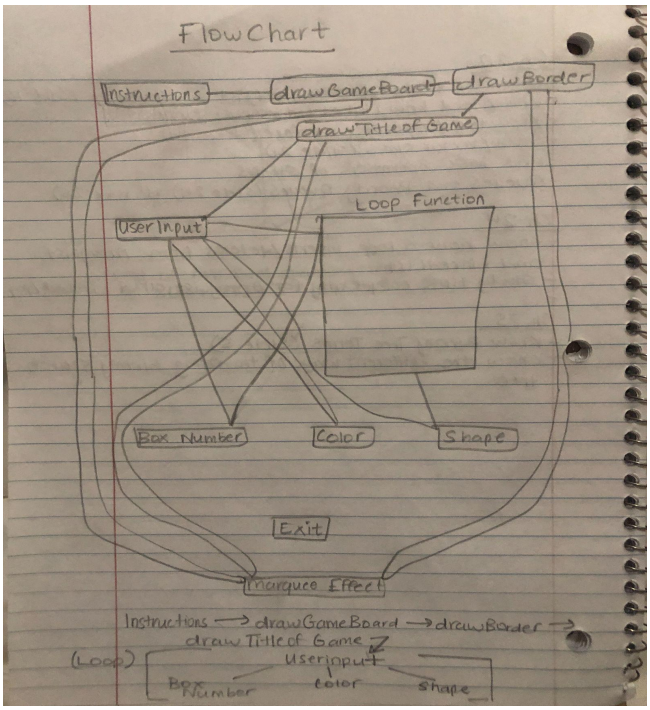
Once this is done, the program will enter the looping section to wait for user input

- The program will ask the first player to enter the box number in which they used
- The user will select one of the box numbers, and the program will jump to that selected box number function and move the starting drawing piece to that box
- The program will now ask the user to select a color
- And the same process will repeat by going to that specific color function and moving that color to the $a2 register that stores the color
- Lastly, the program will ask the first player to select a shape, and the program will once again go to the select a shape function to print that shape in that color in that selected box that the user wanted.
- This process is repeated for Player 2 and will continue until one user wins the game, or until the program is exited by a full game board or by pressing the space bar.

This part of the program is where the data actually flows from one part of the program to another, as the program waits for user input and contains user prompts to make sure the user enters the correct values.

Flowchart:

- Once the Instructions, the gameboard, the border, and the title of the program are printed, the program will go into a big looping that allows data to flow back and forth.
- The program will, at every point, wait for user input for the box number, shape, and color, and will make adjustments accordingly, until the program is exited.

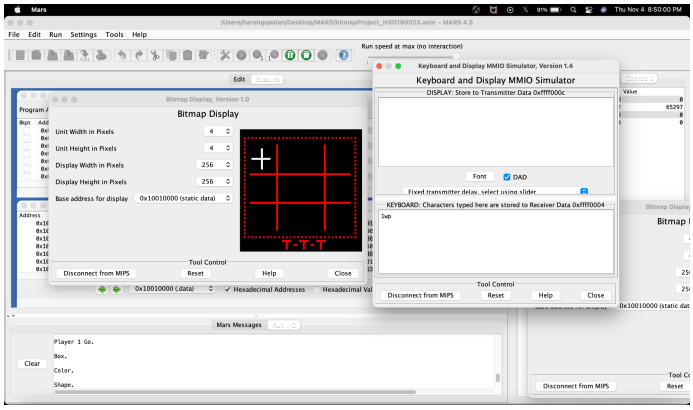


Bitmap Project Report

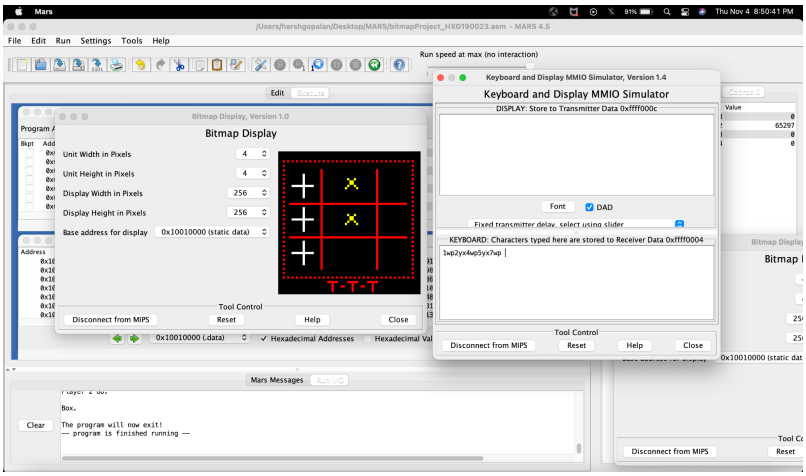
Name: Harsh Gopalan | Class: CS2340. 003

Screenshots of Sample Runs:

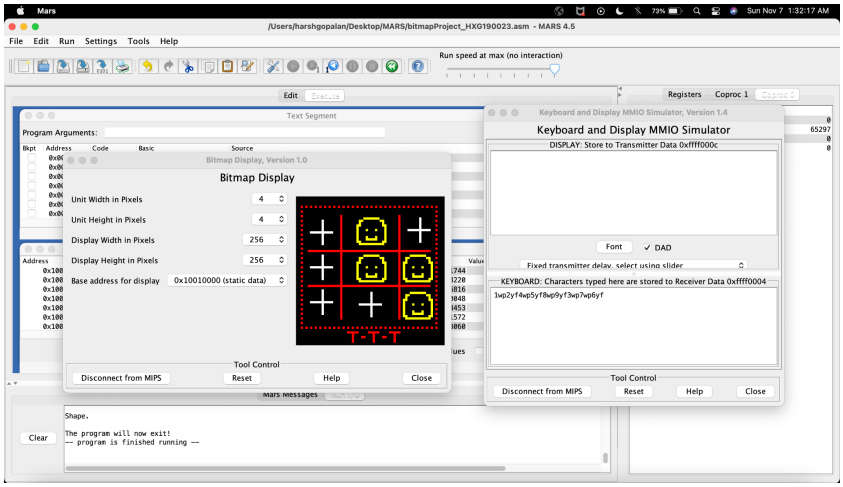
- The following sample run shows how a white plus shape will be drawn at box 1.
- You can also see in the console how it lets the user know what to pick.
- A run-through of this can also be seen in the video run-through.



- In this sample run, you can see how the player with the white plus shape has won the game.
- In this sample run, the program will not let the user know who won. You can press the spacebar to exit the game and re-assemble the program to play again.



- In this sample run, we can see that the gameboard is full. When this happens, the program will exit automatically, given that the users follow the instructions correctly, and do not draw in the same box over again.



Bitmap Project Report

Name: Harsh Gopalan | Class: CS2340. 003

Warnings & Helpful Hints:

Warnings:

- It is necessary that the user enters the box number first, followed by the color, followed by the shape. There should be no spaces between any of these inputs. Looking at the console can help make sure that this error does not happen, as the console will tell exactly what to input.
- The program does not tell who has won the game, but this can be figured out by looking at the gameboard easily.
- The program does not have a check to see if any user decides to change their shape or the shape's color in the middle of the program, so make sure to stick to your choice from the beginning of the game

Helpful Hints:

- Make sure to turn up the volume when running the program, as there are unique sound effects for every shape.
- Watching the run-through video can help by visually seeing the program run.