VIETNAM NATIONAL UNIVERSITY, HO CHI MINH CITY
UNIVERSITY OF TECHNOLOGY
FACULTY OF COMPUTER SCIENCE AND ENGINEERING

COMPUTER ARCHITECTURE (CO2007)

Report

TIC TAC TOE

Advisor: Pham Quoc Cuong

Student Name: Nguyen Thi Ngoc Nhi
Student ID: 2052632

HO CHI MINH CITY, APRIL 2022

Contents

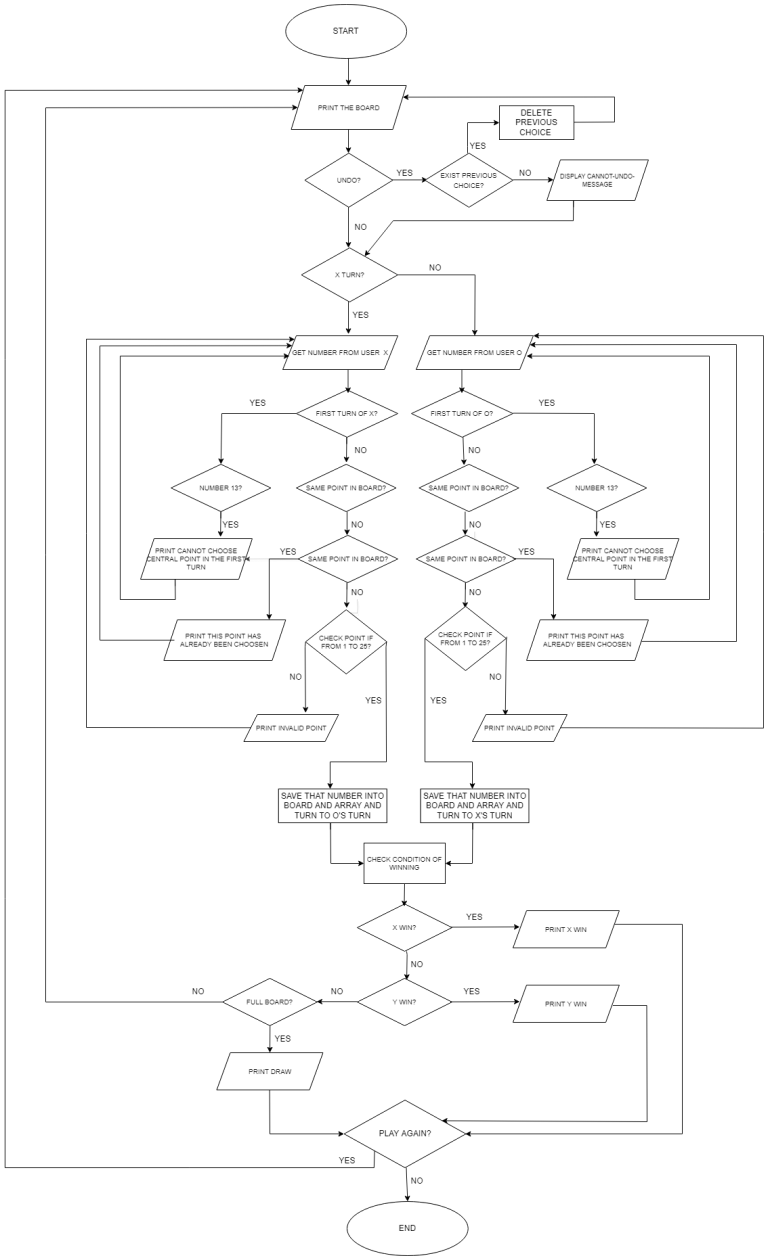
1 Topic

Please design and write MIPS assembly language for implementing a text-based 5x5 board Tic-Tac-Toe game for two players with following requirements.

1. During the first turn of both players, they are not allowed to choose the central point (row 3 & column 3).
2. Any player who has 3 points in a row, column or diagonal will be the winner.
3. Players can undo 1 move before the opponent plays.

2 Explanation

2.1 Using flowchart to visualize the idea



Main idea:
First, we print the board tictactoe 5x5. Then the program will ask the player whether they want to undo or not. If yes, we check if there exists a previous point before, if yes, delete that, else the program will ask the player to enter their first point.
Next, the program will take input from player and then check whether it is 'first move', 'same point in board' or 'invalid point' (these functions will be discussed later). After that, I will check winning condition. If X or O wins, we print that message out. The game is draw when the board is fully-filled. Then the program will ask the player if they want to play again. If yes, we will repeat the game. If no, quit the program.

2.2 First move and Central point

This function will check whether this is player's first move or not.
If this is their first move, we will check if the input number is 13. If yes, we ask the player to enter another point.

I use a variable to count for each time receive the input from player. If that variable is equal to 1, the program will go to check central point

Below is the picture of choosing central point in the first move.

```
---GAME TICTACTOE START ---

########################################
#THE BOARD IS NUMBERIZED FROM 1 TO 25,#
#    LEFT TO RIGHT, TOP TO BOTTOM     #
#    PlayerX will be the first player #
########################################

 1| 2| 3| 4| 5
--+--+--+--+--
 6| 7| 8| 9|10
--+--+--+--+--
11|12|13|14|15
--+--+--+--+--
16|17|18|19|20
--+--+--+--+--
21|22|23|24|25

 | | | |
-+-+-+-+-
 | | | |
-+-+-+-+-
 | | | |
-+-+-+-+-
 | | | |
-+-+-+-+-
 | | | |

Do you want to undo your last choice? Type 1(if yes) or any number(if no): 0

It's playerX's turn
Player X please enter a point: 13

PLAYER CANNOT CHOOSE THE CENTRAL POINT (POINT 13)IN THE FIRST CHOICE
PLEASE ENTER ANOTHER POINT: 1
```

Figure 1: Central point in first move

2.3 Same point in board

This function will check whether the input number has already been in the board or not. If yes, the program will ask the player to input another point. If no, we will save that number into board (for printing) and into the storing-input-number array (for checking winning condition).

First, I create an array full of 25 number zero. Then if which number is visited, I put number one to the position of that number in the array

Below is the picture of player entering an already-chosen-point.

```
 1| 2| 3| 4| 5
--+--+--+--+--
 6| 7| 8| 9|10
--+--+--+--+--
11|12|13|14|15
--+--+--+--+--
16|17|18|19|20
--+--+--+--+--
21|22|23|24|25

 |X| | |
-+-+-+-+-
O| |X|X|
-+-+-+-+-
 |O| | |
-+-+-+-+-
 | | | |
-+-+-+-+-
 | | | |

Do you want to undo your last choice? Type 1(if yes) or any number(if no): 0

It's playerO's turn
Player O please enter a point: 8

This point has already been chosen, please enter another point:
```

Figure 2: Already been chosen point

2.4 Invalid point

This function will check whether the input number is from 1 to 25 or not. If yes, the program will continue the game. If no, the program will ask the player to input another point.

Below is the picture of player entering an invalid point

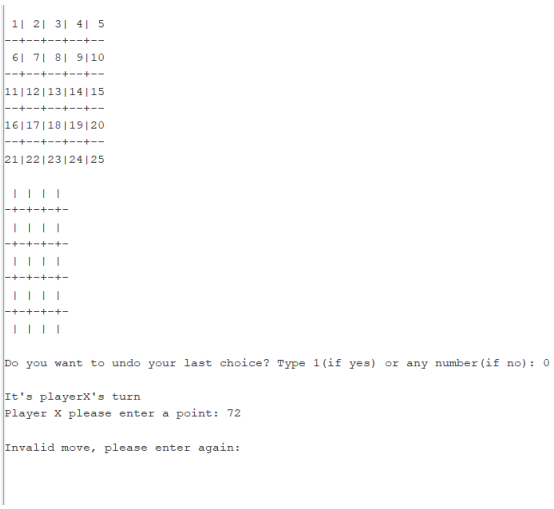


Figure 3: Invalid point

2.5 Undo

This function will check whether there exists any previous move before. If yes, the program will delete that move and print the board and ask the player to input another point. If no, the program will print cannot-undo-message (because there is no previous move) then it will ask the player to enter their first move.

```
 1| 2| 3| 4| 5
--+--+--+--+--
 6| 7| 8| 9|10
--+--+--+--+--
11|12|13|14|15
--+--+--+--+--
16|17|18|19|20
--+--+--+--+--
21|22|23|24|25

 | | | |
-+-+-+-+-
 | | | |
-+-+-+-+-
 | | | |
-+-+-+-+-
 | | | |
-+-+-+-+-
 | | | |

Do you want to undo your last choice? Type 1(if yes) or any number(if no): 1

You cannot undo! You havent entered any point!
It's playerX's turn
Player X please enter a point:
```

Figure 4: Undo at the first turn

2.6 Checking winning condition

Winner is the one who first get 3 marks in a horizontal, vertical or diagonal line
I decided to list all the possible cases of wining. There are total 48 cases (15 for horizontal, 15 for vertical and 18 for diagonal line)

Horizontal

For one row, we have 3 winning cases (figure 5).
In this case, we have 5 rows. So there are 3 x 5 = 15 winning cases for horizontal.

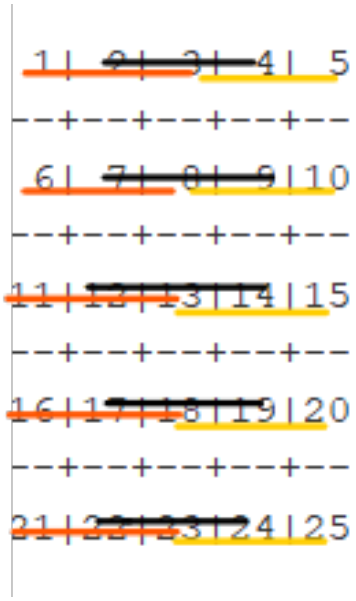


Figure 5: Horizontal winning cases

Vertical

For one row, we have 3 winning cases (figure 6).
In this case, we have 5 rows. So there are 3 x 5 = 15 winning cases for vertical.

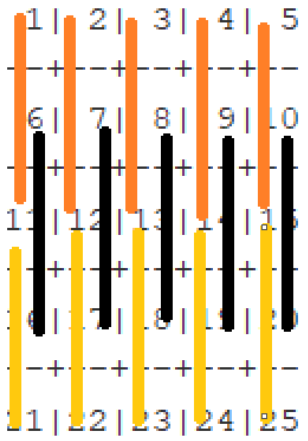


Figure 6: Vertical winning cases

Diagonal

For diagonal, we have 18 cases in the figure 6.

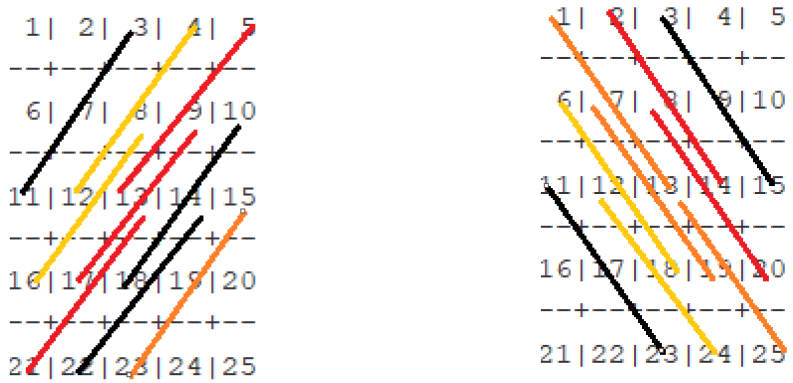


Figure 7: Diagonal winning cases

I create an array to store all the input number from player. If it is from X, I put 1 in array at that postion, if it is from O, I put 2 in array at that postion. Then I use logical instruction AND to check whether it is in 48 cases above or not.

For example, I will check condition for case 1 (point 1, point 2, point 3). If player X marked all the 3 points, then first we use AND instruction for point 1 and point 2 -> we got 1. Then we continue using AND instruction for that result and point 3 -> got 1. If the result is not equal to 0(*), we go to winning function to check whose turn it is to know the winner.

(*) When it comes to player O, we use AND for point 1 and point 2 -> got 2. Then we continue to use AND for that result and point 3 -> we got 2. So, if the result is not equal to 0 -> check winning condition

To know whose turn it is, I use a variable to count the turn. Each turn the variable will be added 1. If the variable is divisible by 2, it is X's turn, else it is O's turn.

	and	
	0	1
0	*0*	*0*
1	*0*	*1*

Figure 8: Logical instruction AND

2.7 Play again

This function will ask player whether they want to play again or not. If yes, the program will start again the game. If no, the program will quit.
Below is the picture of quiting the program

```
 1| 2| 3| 4| 5
--+--+--+--+--
 6| 7| 8| 9|10
--+--+--+--+--
11|12|13|14|15
--+--+--+--+--
16|17|18|19|20
--+--+--+--+--
21|22|23|24|25

 | | |X|
-+-+-+-+-
 | |X|X|
-+-+-+-+-
 |O|O|O|
-+-+-+-+-
 | | | |
-+-+-+-+-
 | | | |

Player O win
 Do you want to play again? Type 1(if yes) or any number(if no):  0

-- program is finished running --
```

Figure 9: Quiting the program

3 Playing Tic Tac Toe

Below is the example of playing Tic Tac Toe game

```
---GAME TICTACTOE START ---

########################################
#THE BOARD IS NUMBERIZED FROM 1 TO 25,#
#    LEFT TO RIGHT, TOP TO BOTTOM     #
#    PlayerX will be the first player #
########################################

 1| 2| 3| 4| 5
--+--+--+--+--
 6| 7| 8| 9|10
--+--+--+--+--
11|12|13|14|15
--+--+--+--+--
16|17|18|19|20
--+--+--+--+--
21|22|23|24|25

 | | | |
-+-+-+-+-
 | | | |
-+-+-+-+-
 | | | |
-+-+-+-+-
 | | | |
-+-+-+-+-
 | | | |

Do you want to undo your last choice? Type 1(if yes) or any number(if no):
```

First, the program wil print the board 5x5. The first board is numerized 1 to 25 for the player to easily find the position. The second board is the board that stored X and O. Then the program will ask whether player wants to undo or not. Then it will go to undo function (discussed before)

```
---GAME TICTACTOE START ---

########################################
#THE BOARD IS NUMBERIZED FROM 1 TO 25,#
#    LEFT TO RIGHT, TOP TO BOTTOM     #
#    PlayerX will be the first player #
########################################

 1| 2| 3| 4| 5
--+--+--+--+--
 6| 7| 8| 9|10
--+--+--+--+--
11|12|13|14|15
--+--+--+--+--
16|17|18|19|20
--+--+--+--+--
21|22|23|24|25

 | | | |
-+-+-+-+-
 | | | |
-+-+-+-+-
 | | | |
-+-+-+-+-
 | | | |
-+-+-+-+-
 | | | |

Do you want to undo your last choice? Type 1(if yes) or any number(if no): 0

It's playerX's turn
Player X please enter a point: 13

PLAYER CANNOT CHOOSE THE CENTRAL POINT (POINT 13)IN THE FIRST CHOICE
PLEASE ENTER ANOTHER POINT:
```

Figure 10: Player X undo their first turn

Next, player X has entered the central point (point 13) in their first turn, so the program ask the player X to enter another point

After player O entered a point, the program check winning condition and then print out the message "Player O win". Next, it will ask whether player want to play again or not. If yes, play again. If no, quit the game.

```
PLAYER CANNOT CHOOSE THE CENTRAL POINT (POINT 13)IN THE FIRST CHOICE
PLEASE ENTER ANOTHER POINT: 4

 1| 2| 3| 4| 5
--+--+--+--+--
 6| 7| 8| 9|10
--+--+--+--+--
11|12|13|14|15
--+--+--+--+--
16|17|18|19|20
--+--+--+--+--
21|22|23|24|25

 | | |X|
-+-+-+-+-
 | | | |
-+-+-+-+-
 | | | |
-+-+-+-+-
 | | | |
-+-+-+-+-
 | | | |

Do you want to undo your last choice? Type 1(if yes) or any number(if no): 0

It's playerO's turn
Player O please enter a point: 14
```

Figure 11: Player O's turn + check winning conditon

```
 1| 2| 3| 4| 5
--+--+--+--+--
 6| 7| 8| 9|10
--+--+--+--+--
11|12|13|14|15
--+--+--+--+--
16|17|18|19|20
--+--+--+--+--
21|22|23|24|25

 | | |X|
-+-+-+-+-
 | | | |
-+-+-+-+-
 | | |O|
-+-+-+-+-
 | | | |
-+-+-+-+-
 | | | |

Do you want to undo your last choice? Type 1(if yes) or any number(if no): 0

It's playerX's turn
Player X please enter a point: 8
```

Figure 12: Player X's turn + check winning conditon

```
 1| 2| 3| 4| 5
--+--+--+--+--
 6| 7| 8| 9|10
--+--+--+--+--
11|12|13|14|15
--+--+--+--+--
16|17|18|19|20
--+--+--+--+--
21|22|23|24|25

 | | |X|
-+-+-+-+-
 | |X| |
-+-+-+-+-
 | | |O|
-+-+-+-+-
 | | | |
-+-+-+-+-
 | | | |

Do you want to undo your last choice? Type 1(if yes) or any number(if no): 0

It's playerO's turn
Player O please enter a point:
```

Figure 13: Player O's turn + check winning conditon

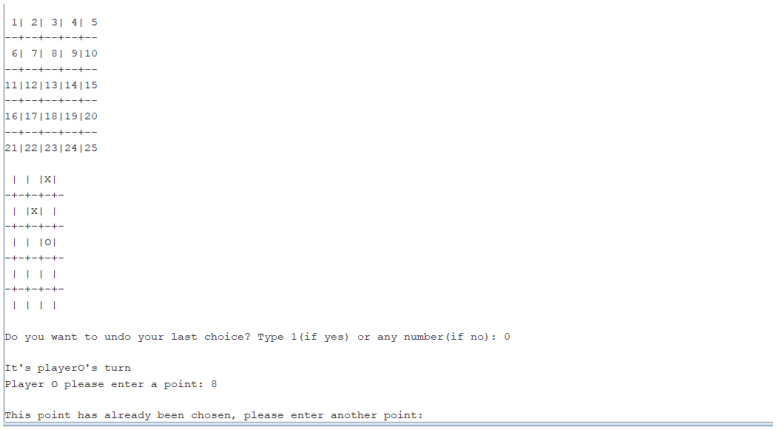


Figure 14: Player O choose a chosen point

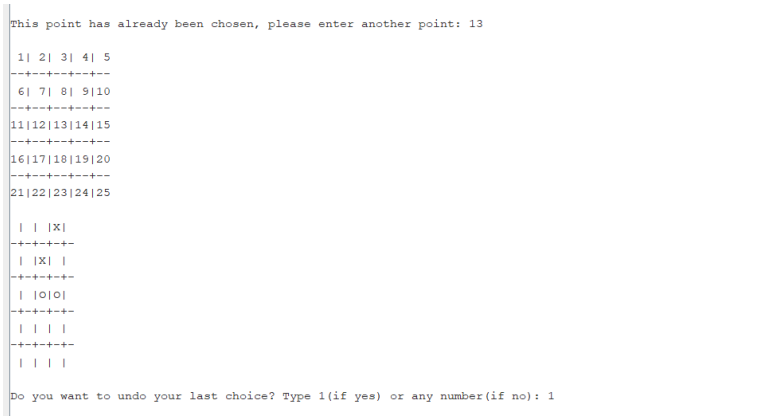


Figure 15: Player O undo their last choice

Figure 16: Player O enter another point after undo their last move

```
It's playerX's turn
Player X please enter a point: 9

 1| 2| 3| 4| 5
--+--+--+--+--
 6| 7| 8| 9|10
--+--+--+--+--
11|12|13|14|15
--+--+--+--+--
16|17|18|19|20
--+--+--+--+--
21|22|23|24|25

 | | |X|
-+-+-+-+-
 | |X|X|
-+-+-+-+-
 |O| |O|
-+-+-+-+-
 | | | |
-+-+-+-+-
 | | | |

Do you want to undo your last choice? Type 1(if yes) or any number(if no): 0

It's playerO's turn
Player O please enter a point:
```

Figure 17: Player X's turn + check winning conditon

```
 1| 2| 3| 4| 5
--+--+--+--+--
 6| 7| 8| 9|10
--+--+--+--+--
11|12|13|14|15
--+--+--+--+--
16|17|18|19|20
--+--+--+--+--
21|22|23|24|25

 | | |X|
-+-+-+-+-
 | |X|X|
-+-+-+-+-
 |O|O|O|
-+-+-+-+-
 | | | |
-+-+-+-+-
 | | | |

Player O win
 Do you want to play again? Type 1(if yes) or any number(if no):  0

-- program is finished running --
```

Figure 18: Player O's turn + check winning conditon

4 Conclusion

In conclusion, through this assignment, I know how to implement the game tic tac toe in MIPS, which helps me get a better understanding of MIPS.