VIETNAM NATIONAL UNIVERSITY, HO CHI MINH CITY
UNIVERSITY OF TECHNOLOGY

Computer Architecture (CO2007) - CC02

Assignment for

Four in a row

Advisor:	Băng Ngọc Bảo Tâm
Student:	Nguyễn Quốc Minh Thư - 2052736.

HO CHI MINH CITY, November 2022

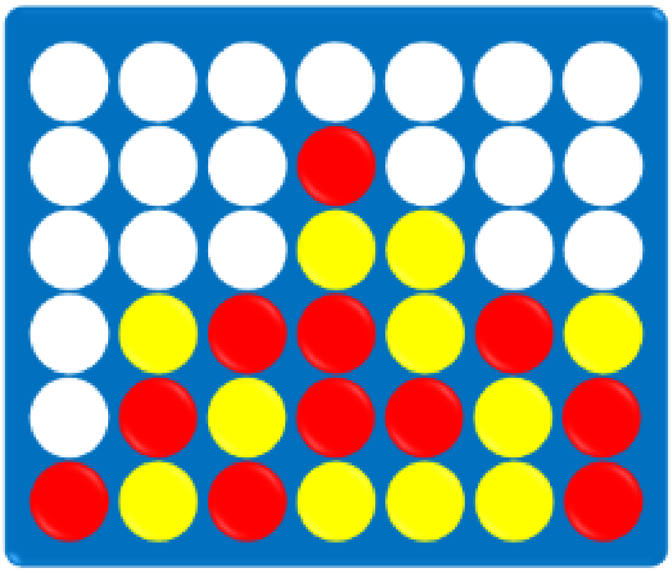
Contents

1 Requirement Specification

Requirement: Design and write MIPS assembly language for implementing a text-based Four in a Row game for two players as follows:

- First, randomly choose the starting player and let this player pick the piece (X or O). The other one has to stick with the remain.
- Then, let the game begin. Four in a Row rules are based on the description.
- Moreover, in the middle of the game (after their first move), each player has 3 times to undo their move (before the opponent's turn).
- Finally, the output of the program is the result of the game.

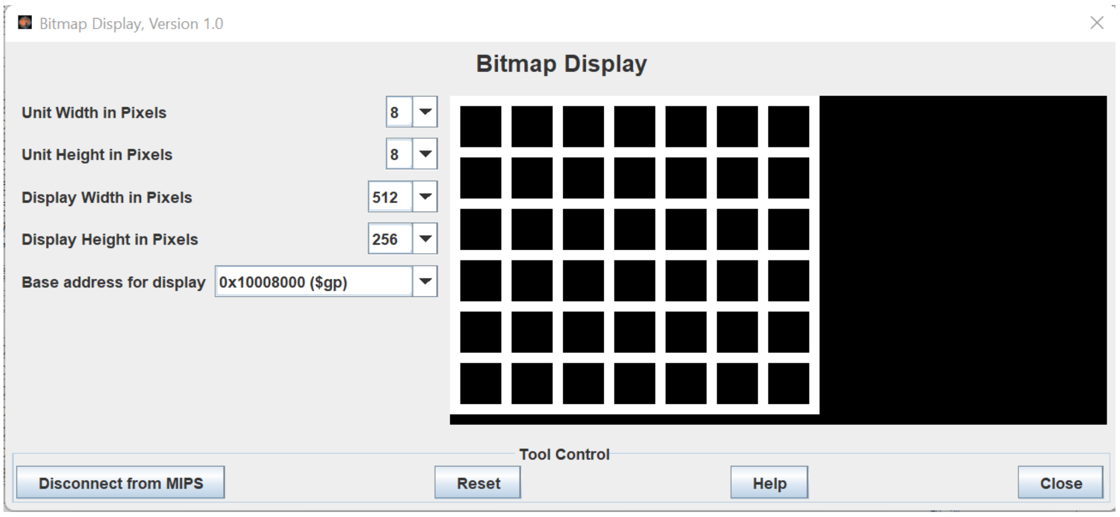
In addition, students have to handle the exception of placing a piece at an inappropriate column by restarting the move. If any players try to violate it 3 times. This player will lose the game.


Hình 1: Four in a row

2 Friendly interface

Description: In this assigment, I choose **Bitmap Display Tool** to display the table and the game's process.

- First, **each units** will have the height and weight equal **8 Pixels**.
- Then, the thickness of the **table boundary line** would be **one unit** (= 8 Pixels)
- Moreover, **each cell** of the table will be **4 units width and 4 units height**.
- Finally, we have **8 vertical and 7 horizontal boundary lines**. Combined with **42 cells in 7 collumns and 6 rows** with will take about 8x36= **288 Pixels width** and 8x31=**248 Pixels height** for the display screen.
- Besides , I choose the address **0x100008000** to be the **base address** of this table.


Bitmap Display, Version 1.0

Bitmap Display

Unit Width in Pixels: 8
Unit Height in Pixels: 8
Display Width in Pixels: 512
Display Height in Pixels: 256
Base address for display: 0x10008000 ($gp)

Hình 2: Bitmap Display

How I draw it :

- **Vertical line:** there will be a needs for 31 units of color white for a line (7 + 6*4 = 31). The first top left unit will have the address is base address. Each unit i have the address (i-1) 's address + (512/8)*4 .The first unit in each vertical line is separate by a space of 5*4=20.
- **Horizontal line:** there will be a needs for 36 units of color white for a line (8 + 7*4 = 36). The first top left unit will have the address is base address. Each unit i have the next address of ist forehead unit. The first unit in each row is separate by a space of 5*512/8*4= 1280.

3 Application implementation

3.1 Started

After draw the white table, I get starting with my Four in a Row game.

```
game:

  jal start

  jal choose_color

  jal gameloop

  j exit
```

Start: Print out the greeting !!!
Choose_color: Function let first player choose their color cell, the rest one will automatically assign to player 2.
Gameloop: Where the game start to receive input and check all the valid base on the rule of the game.

3.2 Get input

Description: Get input of the collumn and return a cell'postion that make sure it drop to the right place. And store it to check later.

Get row from collumn: I create a array for storing the instant row that was not occupied by player yet to store the next drop. And also to check full collumn and full board.

```
.data
array_6: .word 6,6,6,6,6,6,6
```

Get session player and show color corresponding to the current move: After find the position of the cell player access then I call draw

```
.data
array_6: .word 6,6,6,6,6,6,6
```

Store the current move to check later: Create an array with 42 word to store the current move of identified player. And also I create a place to store the instant position of the current move for easier to check later.(interger number of col and row of current cell accessed)

```
.data
board_array: .space 168
instantrow: .word 0
instantcol: .word 0
```

3.3 Rule controlling

Description:

- Get input check valid input (if exception minus one live).
- Let player be able to undo their move 3 times.
- Check win in Vertical, Horizontal, Diagonal.

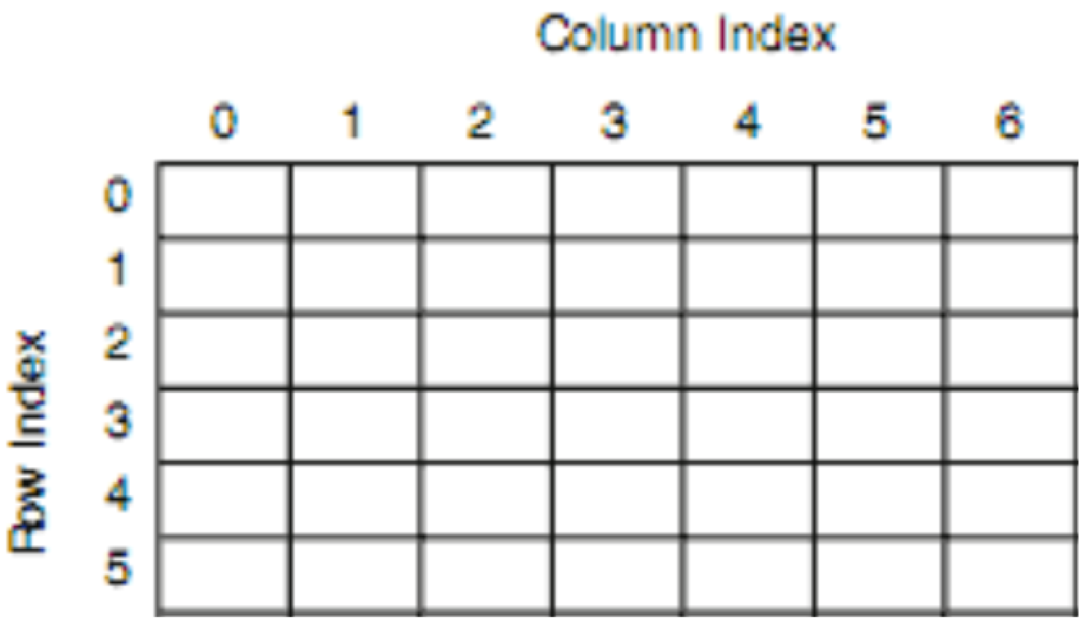
Deal with exception move: let give player 3 live, and if they violate with exception move their live will be minus. Check live after each exception move minus.

```
,data
live_1: .word 3
live_2: .word 3
```

Deal with undo move: let give player 3 times to undo, and if they use them all, then won't show the option of undo anymore. Give the ask right after they enter the input collumn.

```
,data
undo_1: .word 3
undo_2: .word 3
```

Check win: My 7 collums , 6 rows table will have index as below.



Hình 3: Bitmap Display

Chech win vertical: Idea here is to get the current move of player then check from the current row in that current column down to the bottom of table.

```
    #Pseudo code
    checkwincol(){
if(player1) instant_session_value=1;
if(player2) instant_session_value=2;

instant_cell.col= instant_col;
instant_cell.row= instant_row;

count=0;

// because we drop it down so we just check from the current row down
for (int i=0; i<(6-instant_cell.row); i++){
    a=get_value_inside_cell(instant_cell.col, instant_cell.row +i );
    b= instant_session_value;
    if(a==b){
       count++;
       Continue;
    }
    count=0;
}
if(count==4) win(Player);
return;
}
```

Chech win Horizontal: Idea here is to get the current move of player then check all the cell in the current row.

```
    #Pseudo code
    checkwinrow(){
if(player1) instant_session_value=1;
if(player2) instant_session_value=2;

instant_cell.col= instant_col;
instant_cell.row= instant_row;

count=0;

// because we drop it down so we just check from the current row down
for (int i=0; i< 7; i++){
    a=get_value_inside_cell(instant_cell.col +i, instant_cell.row  );
    b= instant_session_value;
    if(a==b){
       count++;
       Continue;
    }
    count=0;
}
if(count==4) win(Player);
return;
}
```

Chech win digonal: Idea here is to get the current move of player then check two slash (taybacdongnam and dongbactaynam) and with 2 dimension in each slash

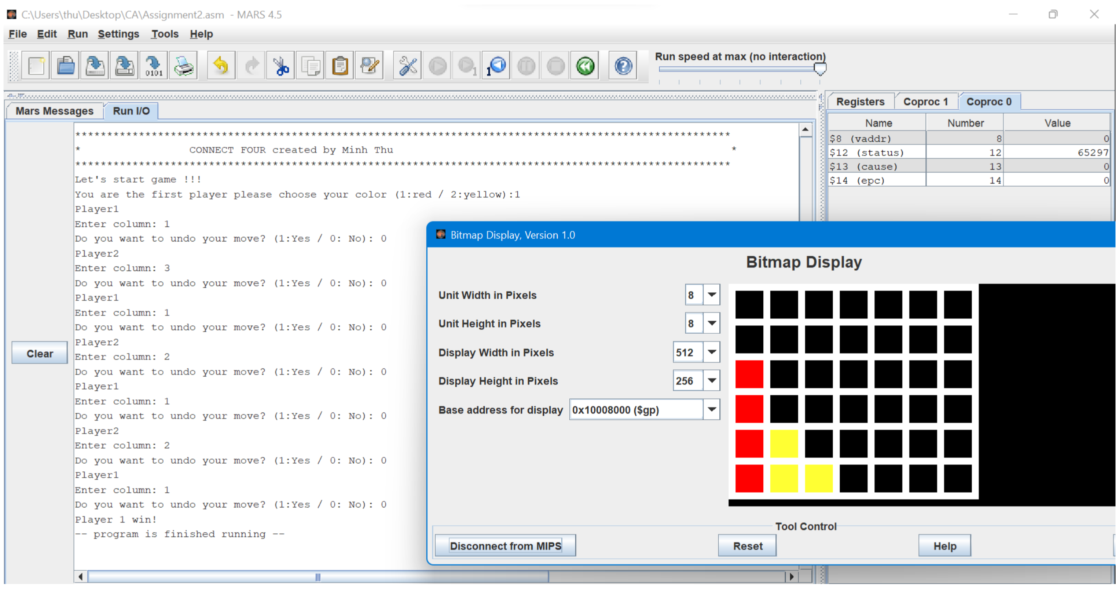
```
    #Pseudo code
    checkwindigonal(){

    \\ First slash
    checktaybacdongnam();

    \\ Second slash
    checkdongbactaynam();

}
```

Taybacdongnam Slash Idea:

```
    #Pseudo code

    checktaybacdongnam(){
    if(player1) instant_session_value=1;
if(player2) instant_session_value=2;

instant_cell.col= instant_col;
instant_cell.row= instant_row;

count=0;

// check up not including the instant cell

for (int i=0; i< min(instant_cell.row,instant_cell.col); i++){
    a=get_value_inside_cell(instant_cell.col-i-1, instant_cell.row -i-1 );
    b= instant_session_value;
    if(a==b){
       count++;
       Continue;
    }
   break;
}
// check down including the instant cell

for (int i=0; i< 6 - max(instant_cell.row,instant_cell.col); i++){
    a=get_value_inside_cell(instant_cell.col+i, instant_cell.row +i );
    b= instant_session_value;
    if(a==b){
       count++;
       Continue;
    }
    break;
}
if(count==4) win(Player);

return;
}
```

4 Demo

1. Simple test with win vertical case.



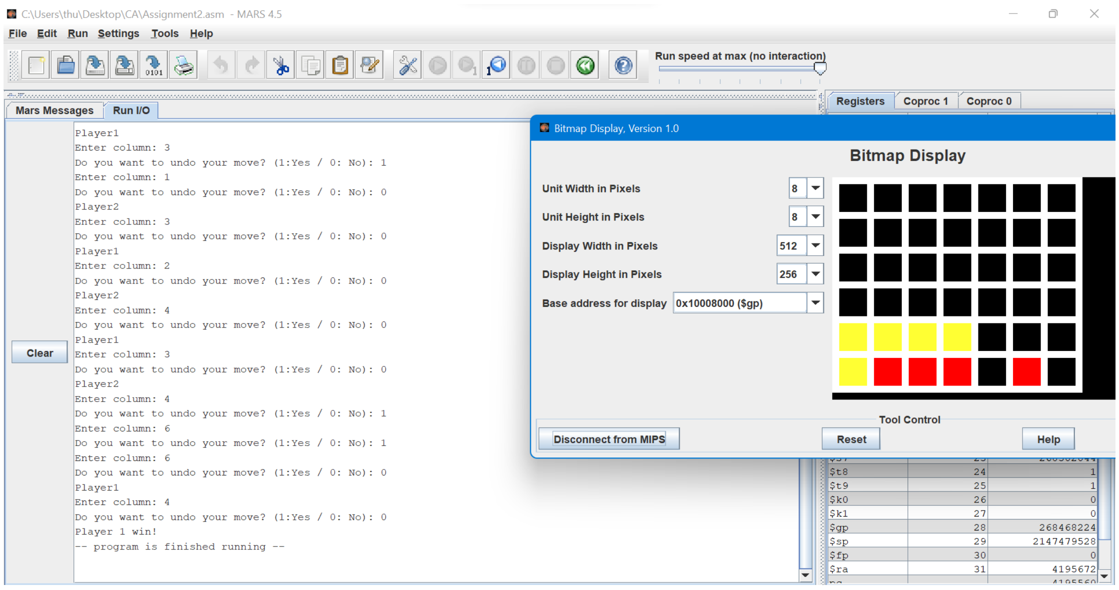
```
*******************************************************************************************
*                 CONNECT FOUR created by Minh Thu                                         *
*******************************************************************************************
Let's start game !!!
You are the first player please choose your color (1:red / 2:yellow):1
Player1
Enter column: 1
Do you want to undo your move? (1:Yes / 0: No): 0
Player2
Enter column: 3
Do you want to undo your move? (1:Yes / 0: No): 0
Player1
Enter column: 1
Do you want to undo your move? (1:Yes / 0: No): 0
Player2
Enter column: 2
Do you want to undo your move? (1:Yes / 0: No): 0
Player1
Enter column: 1
Do you want to undo your move? (1:Yes / 0: No): 0
Player2
Enter column: 2
Do you want to undo your move? (1:Yes / 0: No): 0
Player1
Enter column: 1
Do you want to undo your move? (1:Yes / 0: No): 0
Player 1 win!
-- program is finished running --
```

Name	Number	Value
$8 (vaddr)	8	0
$12 (status)	12	65297
$13 (cause)	13	0
$14 (epc)	14	0

Unit Width in Pixels: 8
Unit Height in Pixels: 8
Display Width in Pixels: 512
Display Height in Pixels: 256
Base address for display: 0x10008000 ($gp)

Hình 4: Demo

2. Simple test with win horizontal case.



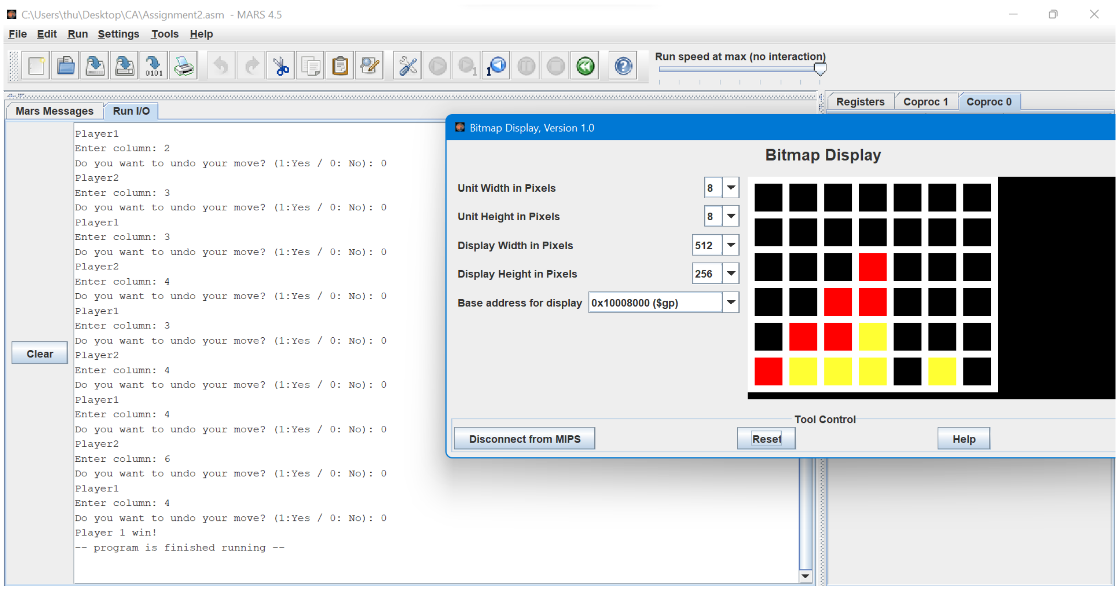
```
Player1
Enter column: 3
Do you want to undo your move? (1:Yes / 0: No): 1
Enter column: 1
Do you want to undo your move? (1:Yes / 0: No): 0
Player2
Enter column: 3
Do you want to undo your move? (1:Yes / 0: No): 0
Player1
Enter column: 2
Do you want to undo your move? (1:Yes / 0: No): 0
Player2
Enter column: 4
Do you want to undo your move? (1:Yes / 0: No): 0
Player1
Enter column: 3
Do you want to undo your move? (1:Yes / 0: No): 0
Player2
Enter column: 4
Do you want to undo your move? (1:Yes / 0: No): 1
Enter column: 6
Do you want to undo your move? (1:Yes / 0: No): 1
Enter column: 6
Do you want to undo your move? (1:Yes / 0: No): 0
Player1
Enter column: 4
Do you want to undo your move? (1:Yes / 0: No): 0
Player 1 win!
-- program is finished running --
```

$t8	24	1
$t9	25	1
$k0	26	0
$k1	27	0
$gp	28	268468224
$sp	29	2147479528
$fp	30	0
$ra	31	4195672

Unit Width in Pixels: 8
Unit Height in Pixels: 8
Display Width in Pixels: 512
Display Height in Pixels: 256
Base address for display: 0x10008000 ($gp)

Hình 5: Demo

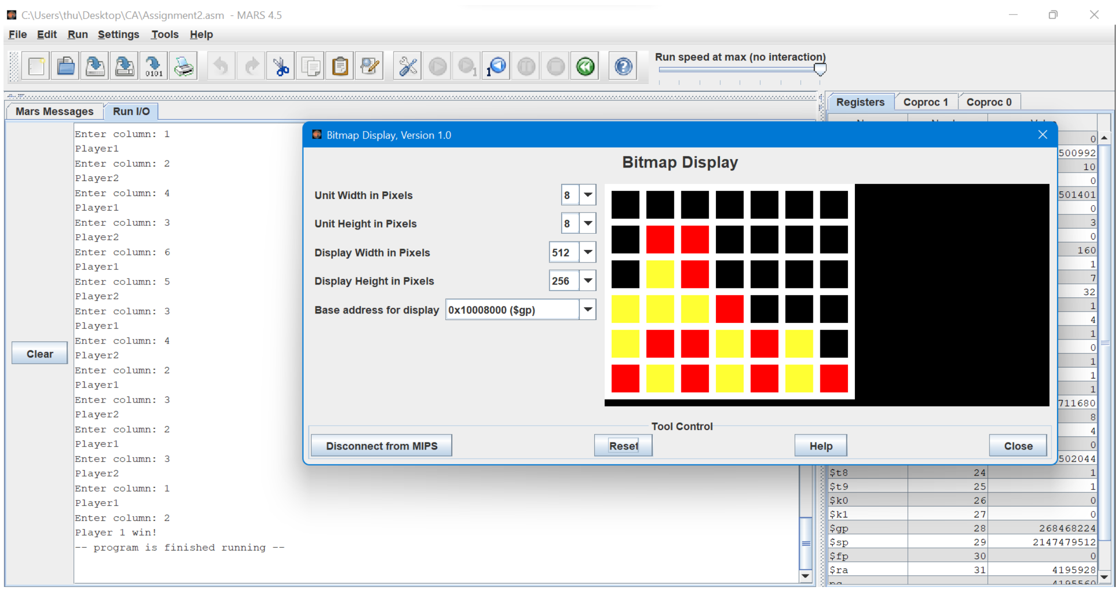
3. Simple test with win diagonal case.



```
Player1
Enter column: 2
Do you want to undo your move? (1:Yes / 0: No): 0
Player2
Enter column: 3
Do you want to undo your move? (1:Yes / 0: No): 0
Player1
Enter column: 3
Do you want to undo your move? (1:Yes / 0: No): 0
Player2
Enter column: 4
Do you want to undo your move? (1:Yes / 0: No): 0
Player1
Enter column: 3
Do you want to undo your move? (1:Yes / 0: No): 0
Player2
Enter column: 4
Do you want to undo your move? (1:Yes / 0: No): 0
Player1
Enter column: 4
Do you want to undo your move? (1:Yes / 0: No): 0
Player2
Enter column: 6
Do you want to undo your move? (1:Yes / 0: No): 0
Player1
Enter column: 4
Do you want to undo your move? (1:Yes / 0: No): 0
Player 1 win!
-- program is finished running --
```

Unit Width in Pixels: 8
Unit Height in Pixels: 8
Display Width in Pixels: 512
Display Height in Pixels: 256
Base address for display: 0x10008000 ($gp)

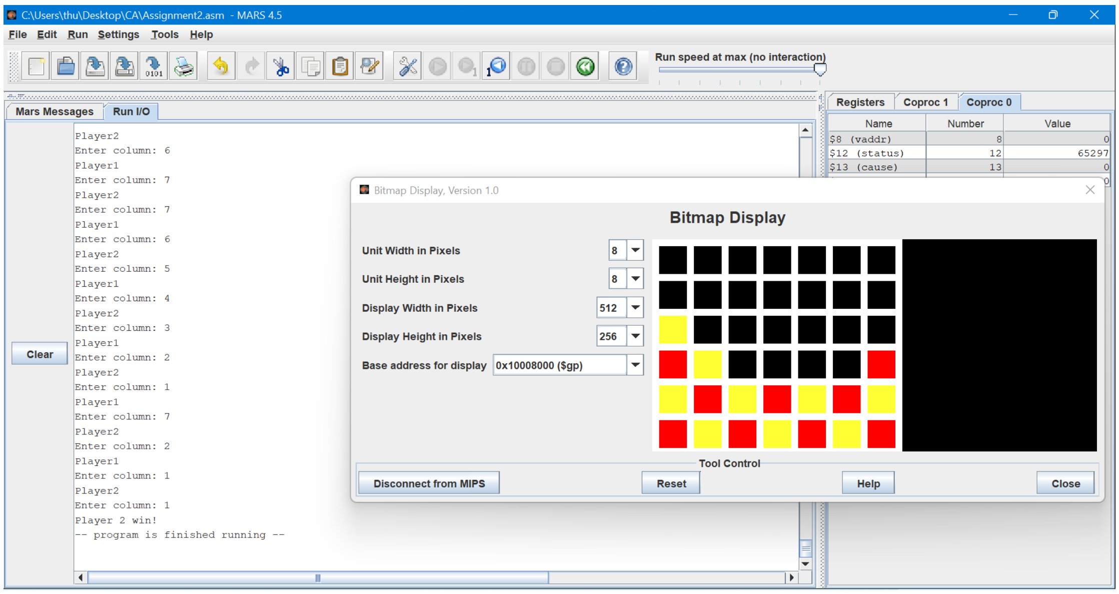
Hình 6: Đông bắc tây nam



```
Enter column: 1
Player1
Enter column: 2
Player2
Enter column: 4
Player1
Enter column: 3
Player2
Enter column: 6
Player1
Enter column: 5
Player2
Enter column: 3
Player1
Enter column: 4
Player2
Enter column: 2
Player1
Enter column: 3
Player2
Enter column: 2
Player1
Enter column: 3
Player2
Enter column: 1
Player1
Enter column: 2
Player 1 win!
-- program is finished running --
```

Unit Width in Pixels: 8
Unit Height in Pixels: 8
Display Width in Pixels: 512
Display Height in Pixels: 256
Base address for display: 0x10008000 ($gp)

Hình 7: Tây bắc đông nam


C:\Users\thu\Desktop\CA\Assignment2.asm - MARS 4.5

```
Player2
Enter column: 6
Player1
Enter column: 7
Player2
Enter column: 7
Player1
Enter column: 6
Player2
Enter column: 5
Player1
Enter column: 4
Player2
Enter column: 3
Player1
Enter column: 2
Player2
Enter column: 1
Player1
Enter column: 7
Player2
Enter column: 2
Player1
Enter column: 1
Player2
Enter column: 1
Player 2 win!
-- program is finished running --
```

```
Name         Number   Value
$8  (vaddr)       8       0
$12 (status)     12   65297
$13 (cause)      13       0
```

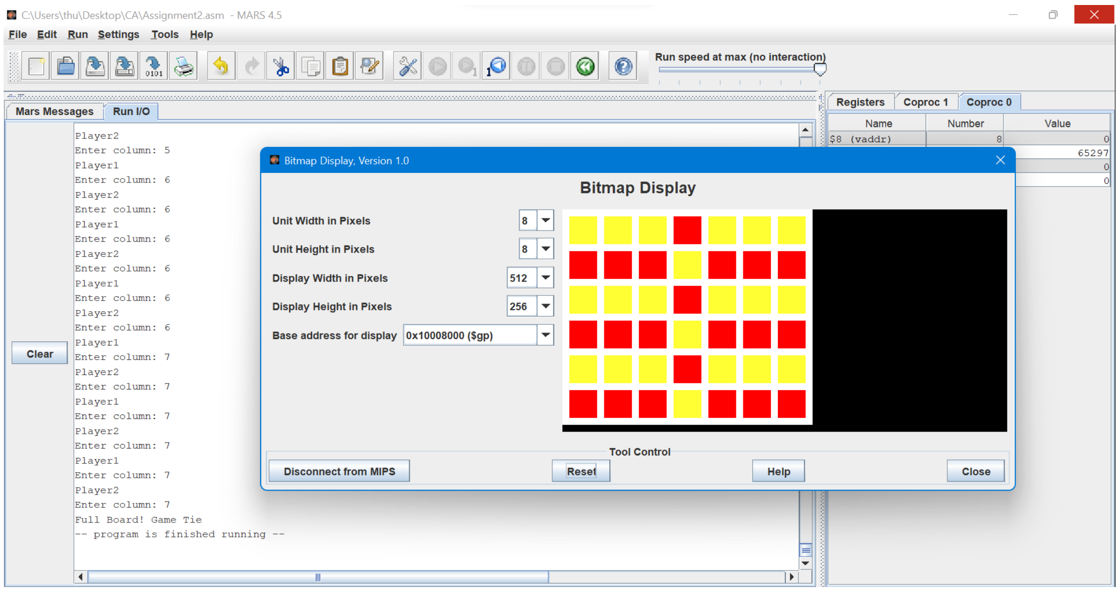
Bitmap Display, Version 1.0

Bitmap Display

Unit Width in Pixels: 8
Unit Height in Pixels: 8
Display Width in Pixels: 512
Display Height in Pixels: 256
Base address for display: 0x10008000 ($gp)

Hình 8: Colum index 0 be the last move

4. Other case


C:\Users\thu\Desktop\CA\Assignment2.asm - MARS 4.5

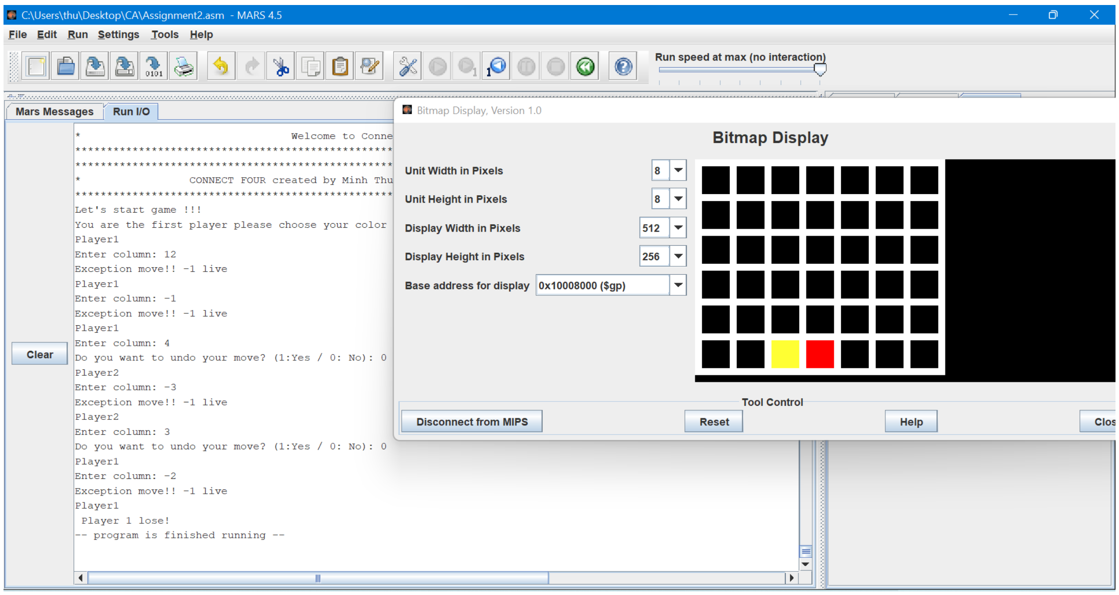
```
Player2
Enter column: 5
Player1
Enter column: 6
Player2
Enter column: 6
Player1
Enter column: 6
Player2
Enter column: 6
Player1
Enter column: 6
Player2
Enter column: 6
Player1
Enter column: 7
Player2
Enter column: 7
Player1
Enter column: 7
Player2
Enter column: 7
Player1
Enter column: 7
Player2
Enter column: 7
Full Board! Game Tie
-- program is finished running --
```

Bitmap Display, Version 1.0

Bitmap Display

Unit Width in Pixels: 8
Unit Height in Pixels: 8
Display Width in Pixels: 512
Display Height in Pixels: 256
Base address for display: 0x10008000 ($gp)

Hình 9: Full board


C:\Users\thu\Desktop\CA\Assignment2.asm - MARS 4.5
File Edit Run Settings Tools Help
Run speed at max (no interaction)
Mars Messages
Run I/O
Clear
Bitmap Display, Version 1.0
Bitmap Display
Tool Control
Disconnect from MIPS
Reset
Help

```
*                                  Welcome to Conne
**************************************************
**************************************************
*                 CONNECT FOUR created by Minh Thu
**************************************************
Let's start game !!!
You are the first player please choose your color
Player1
Enter column: 12
Exception move!! -1 live
Player1
Enter column: -1
Exception move!! -1 live
Player1
Enter column: 4
Do you want to undo your move? (1:Yes / 0: No): 0
Player2
Enter column: -3
Exception move!! -1 live
Player2
Enter column: 3
Do you want to undo your move? (1:Yes / 0: No): 0
Player1
Enter column: -2
Exception move!! -1 live
Player1
 Player 1 lose!
-- program is finished running --
```

Unit Width in Pixels: 8
Unit Height in Pixels: 8
Display Width in Pixels: 512
Display Height in Pixels: 256
Base address for display: 0x10008000 ($gp)

Hình 10: Lose due to exception move