DESIGN AND IMPLMENTATION OF A FPGA ELECTRONIC REVERSI GAME

[1]CHIH-YILIN, [2]CHAI-CHUAN KUO, [3]CHAO-HUANG WEI

[1,2,3]Electrical Engineering Department Southern,
Taiwan University of Science and TechnologyTaiwan
Email: [1]4a12c055@stust.edu.tw, [2]4a12c078@stust.edu.tw, [3]drwei@mail.stust.edu.tw

Abstract - This paper describes a development of an electronic reversi game, which is constituted with a matrix LED as the game board, wherein the discs of both players are represented by red and blue LEDs independently, during the game the LED will be changed to opposite color to represent the flip of discs. The player uses a touch panel to move the disc, and confirm the location to place the disc. The microprocessor will check the legal position of the disc, calculate the players discs, then display the scores on a 7-segment LED immediately.
An hardwired Artificial Intelligence algorithm in the board game will be explored as well. The AI player employs a minimax algorithm to decide the disc move with a heuristic evaluation functions. Certainly, some improvements can be implemented like selection of other game strategy and game difficulty in future works.

Keywords: Reversi, FPGA, Artificial Intelligent, Matrix LED Board.

I. INTRODUCTION

Reversi or Othello is a strategy board game for two players.The modern Reversi was invented in 1883 by British, and was introduced to Japanin 1971 by the Japanese board game expert Mr. Goro Hasegawa.According to his investigation in2005, there are about 60 million players in Japan, (about 15 million players for 'JapaneseShogi', 5 million players for 'GO', and 3 million players for 'international chess').

This game is so popular owing to the simplicity of the game rule. It is suitable for anybody over 6 years old, and the game can be finished within 10 to 60 minutes.Howeverthis game has highly strategycomponent without any lucky chance. Players need technique of observation and strategy planningto win the game.

The game board has 8 x 8 grids,the squares on the reversi board are referred to using coordinate notation in order to record games and allow the discussion of strategy.At the beginning of the game there are 4 discs placed in the center of the board (**Fig. 1a**), the player has black disc can placedisc first, where the X locations denote the allowable positions for black player, then in alternative way to place disc each player.When the opponent's disc is enclosed by current player's disc in horizontal, vertical, and diagonal direction, all the opponent's discs will be flipped to current player's (i.e. change disccolor). When no more locations can be placed the game is over, then by counting discs of both players' the player with mostdiscs is the winner (**Fig. 1b**).

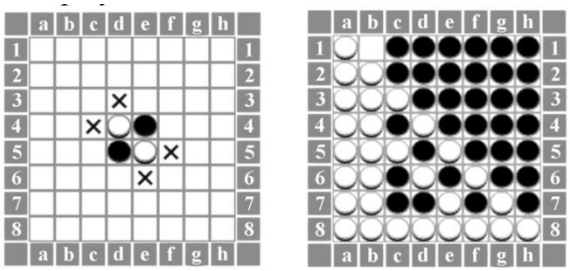


Fig. 1: Starting position

The electronic reversi game was developed in the middle 80 in Nintendo'sfirst arcade game machine and Apple II personal computer, and was later ported to a dedicated home game console - Game Boy.Due to the performance of computer increases exponentially since 1990, the best computer programs have easily defeated the best human playersnowadays. Analysts have estimated the number of legal positions in reversi is at most 10^{28}, and it has a game-tree complexity of approximately 10^{58}, that is quite difficult for humansto lookahead.

In this paper the reversi game will be implemented with a software and hardware integrated embedded system. The essential part is to design a parallel processing unit in hardware instead of sequential procedure in software for artificial intelligent algorithm.

II. MOTIVATIONAND OBJECTS

2.1. Materials and Procedures

The reversi is a popular two player strategy board game inmany countries. There are numerous players around the world especially in Asia. The game isplayed on an 8×8 unchecked boardusually, it ends if

all squares are occupied or if none ofthe players can take a legal move on the remaining emptyboard squares. In this game mode it is quite trivial to realize this system using the software procedure in **Fig. 2**.The computer executesmainlythe functions to display the player's discs on an LED board, checks the legal position of disc move, and calculates scores of both players.

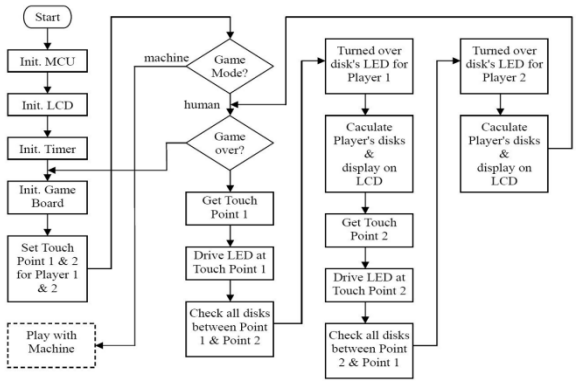


Fig. 2: Software procedure of reversi for human players

When playing with the machine, we use anartificialintelligent as an avatar of the opponent.

2.2. Considerations of electronic game

An electronic game is a game that employs electronics to create an interactive system with which a player can play. The most common form of electronic game today is the video game, which uses a screen to display the game images and some information relative to the game status.These games involve normally interaction with a user interface to generate visual feedback on a display device. Therefore they are not only for fun, but also quite suitable as test tools for development of human-machine interface.

One of the essential part of the electronic game is its intelligent, when human plays with the machine, it is a big challenge for human to give the machine autonomous operations, how good it will be. The techniquesinclude algorithms in control theory, robotics, computer graphics and computer science.Game playing was an area of research in AI from its beginning, many problemslike incomplete information, pathfinding, real-time decisions, and economic planning need to be solved in the real-time strategy games.

The purpose of this project is not only to realize the reversi game on a FPGA board, but also provide an implementation of the AI algorithm with a hardware circuit, therefore reduces the processing time of the computing intensive software procedure dramatically.

2.3. System Implementation

Based on the above descriptions, this reversi game should include at least a user interface, a display device and an AI processing unit. The systemarchitecture will be developed as shown in following figure, which includes mainly a reversi hardware module, a matrix LED controller, a LTM touch screen controller, a VGA display controller, anembedded softcoreNios II CPU and some peripheral components(**Fig. 3**).

2.3.1Nios II processor

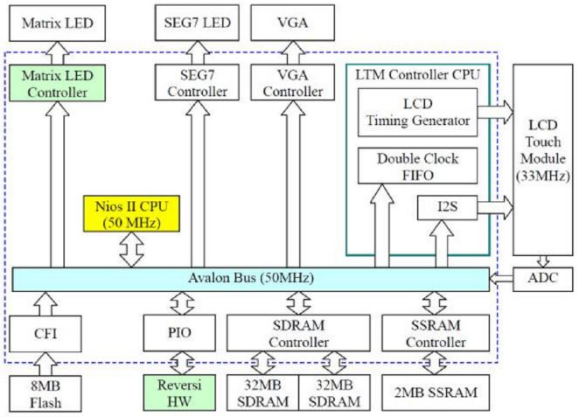


Fig.3:Function blocks of electronic reversi game

The 32-bit Altera's Nios II processor in this embedded system is the most widely used soft processor in the FPGA industry(**Fig. 4**). It delivers exceptional flexibility for cost-sensitive, real-time, safety-critical, ASIC-optimized, and applications processing requirements. The maximum performance efficiency of Nios II processorvaries from 0.15 to 1.13 MIPS per MHzaccording to core architecture, while the ARM Cortex-A9 MPCorecan reach up to 2.5MIPS/MHz.But with fusing of hardware circuit, the Nios II can become a very high performance system especially for real time applications.

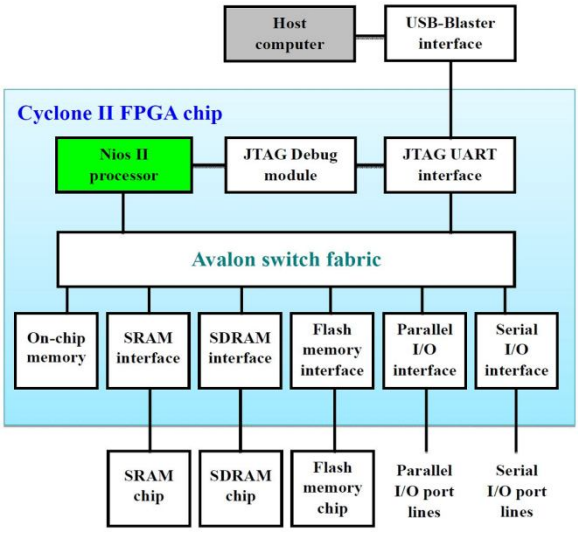


Fig. 4:A Nios II system implemented on the DE2 board

Altera offers a free embedded software toolsalso to develop robust software applications. The Eclipse-based Nios II software build tools for Eclipse Nios II EDS includesa fully integrated graphical development environment, GNU tools, Software

examples and templates,TCP/IP Network Stack, and MicroC/OS-II RTOS.
Since the most computing intensive AI algorithm were implemented with hardware circuit, the Nios II CPU plays a vital role as a system manger, controls data transfer between peripherals mainly.

2.3.2Matrix LED game board

The game board in this project is realized by an 8 x 8 matrix RGB LED, as conventional way besides the power supply, at least 32 control lines are required(8 rows, 8 columns x 3 RGB) for this board. To saving the complexity of the control circuitry, an intelligent control LED light source (WS2812B) , which is a RGB LED chip with integrated control circuit ina 5050package was used (**Fig. 5**). It includesa serial digitaldata latch portand a signal reshaping amplification driving circuit for cascading signal transmissionby single line. Each pixel of the three primary color can achieve 256 highly consistentbrightness, completed 16 million full color display.

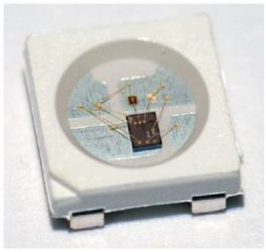
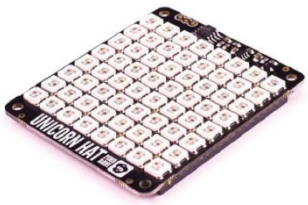



Fig.5:WS2812B RGB LED and 8x8 matrix LED board

The data transfer protocoluses single NZR communication mode.After the pixelpower-on resets, the DIN port receivesdata from the controller, the first pixelcollects initial 24bit data then sends it to the internal data latch.Theother serial data, which is reshaped by the internal signal reshaping amplification circuit, is sent to the next cascade pixelthrough the DO port (**Fig. 6**).

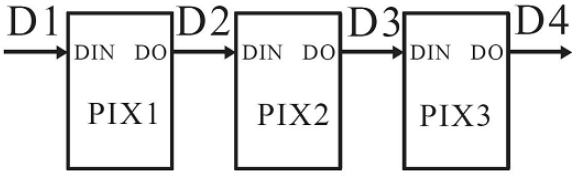


Fig. 6:Cascaded one-wire LED communication interface

Each bit is represented by a 1.25us long (800KHz) high-low pattern with a 1⁄3:2⁄3 (2⁄3:1⁄3) duty cycle for a 0-value (1-value) (**Fig. 7**, **Tab. 1**).

Tab. 1:Definition of pixel data timing

T0H	0 code, high voltage time	400ns	±150ns
T0L	0 code, low voltage time	850ns	±150ns
T1H	1 code, high voltage time	800ns	±150ns
T1L	1 code, low voltage time	450ns	±150ns
RET	low voltage time	Above 50μs	

Data transfer time(TH+TL=1.25μs±600ns)

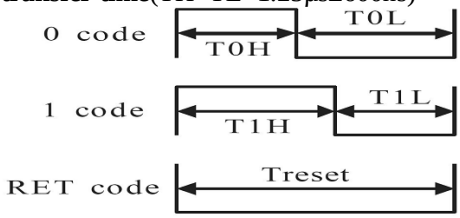


Fig.7: Waveform of '0' and '1' signal

Typically bit-banging an I/O line is the common method of driving the WS2812B RGB LEDs using a microcontroller. However, to fit with the signal timing requirement of the WS2812B this will ended up being very resource intensive. A better method is to use anextra hardware Serial Peripheral Interface (SPI) module to handle transferring data, thatshifts out the pixel data over the Serial Data Out (SDO) pinwithout CPU intervention.
SPI has an integrated shift register, which shifts a bit out with every clock cycle.The implementation is to convert one bit of LED data to one byte of SPI data, for example, a '1' to 0xF8 and a '0' to 0xE0. The hexadecimal code has 2⁄3 (1⁄3) high time and 1⁄3 (2⁄3) low time for 0xF8 (0xE0). It's quite an easy implementation, but on the downside it also needs a high SPI clock to reach the 800KHz for the one-wire LED data stream(i.e. 8x800KHz = 6.4MHz for SPI clock), and an even higher CPU clock to bring the data to the SPI module. Therefore it needs to pre-calculate, store the result in a buffer and transmit afterwards, thatneeds 8 times the memory buffer (i.e. 24bytes per LEDinstead of 24 bits raw LED data).

2.3.3LTM Touch Panel

There is a 4.3"LCD Touch Panel Module (LTM)on the DE2-70 development board, whichprovides an 800x480 full-color high-quality TFT LCD display.The image to be displayed were pre-stored in the on board flash ROM. Since there are no frame buffer and no self refreshcontrol circuit in the LCD module, the required controller should be realized in the FPGA chip.
To simplify the image processing, all images stored in the flash ROM were not compressed. They are exactly size of 800x480 pixels in 24bit BMP data format. **Fig. 8**showsthree pre-stored images for this project.



Fig. 8:Pre-stored images of reversi game

The LTM is equipped with an AD7843 touch screen digitizer chip from Analog Devices. The AD7843 is a 12-bit analog to digital converter (ADC) for digitizing x and y coordinates of touch points applied to the touch screen. When the panel is touched via a pen or a finger, an interrupt signal indicates this event, then the coordinates of the touch point stored in the AD7843 can be obtained by a serial port interface.**Fig. 9** shows the x/y coordinate values of the 4 corners on the touch panel. Other touch points on the panel can be calculated using interpolation method in vertical and horizontal directions.

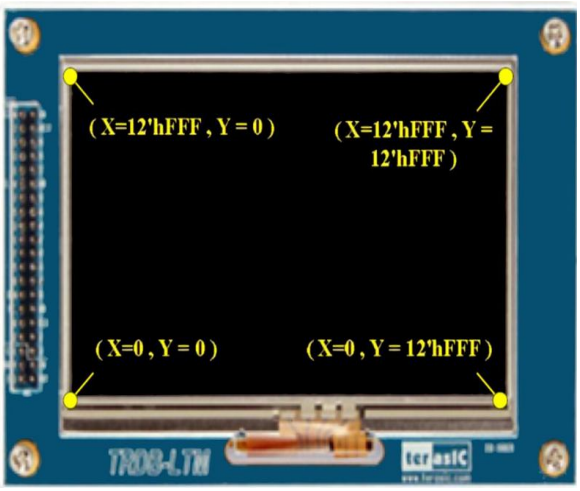


Fig. 9:Coordinates of the corners on the LTM touch screen

2.4Game Strategy

In the classic game of reversiplayers will try to end up with the most discs of their colorat the end of the game.Because the peculiar game rule, it is quite easy reverse the scores dramatically during the game, even in the end phase of the game a lot of opponent's discs can be flipped with only a few moves. Therefore it is wise to consider the best position to flip opponent's discs instead of worryingabout how many the player's discshave.

On the other hand, it is also quite easy to lose control of the game in the first few moves. Play the wrong move and the opponent will be able to control the subsequence of the rest game to their favors.But both player have the same legal positions for selection at first step.

The center positionsare most being attacked by horizontal, vertical, and diagonal directions,on the other hand,the position of the four corners is completely impossible to be attacked. The edge positions are next to the 4 corners, which can be attackedby onedirectiononly.The final phase of the game is to gain key positions, while the pre-placement is a preparation to seize key positions.For example,the player should avoid place disc on the position b2, b7, g2, and g7 to prevent the opponent occupy the valuable corner position.**Fig. 10** shows therisk regions on the board, regions 1 and 2 are very valuable strategic areas, in contrast, regions 5 and 4 are risky territory.

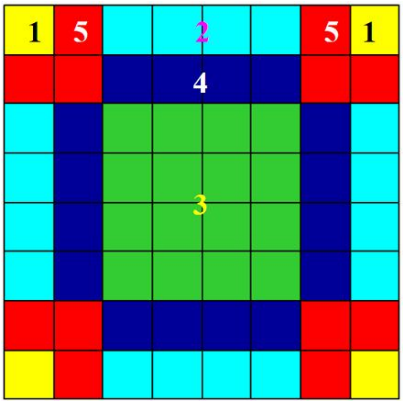


Fig. 10:Risk Regions on board

The computer evaluatestheweighting value associated with each position on the game board to be used for AI algorithm. **Fig. 11**illustrates the value assignments of one eighth of this game board, same assignments are for the rest positions. It should be readily apparent that once a disc has been placed on a corner square it cannot be flipped, and thus has most value.

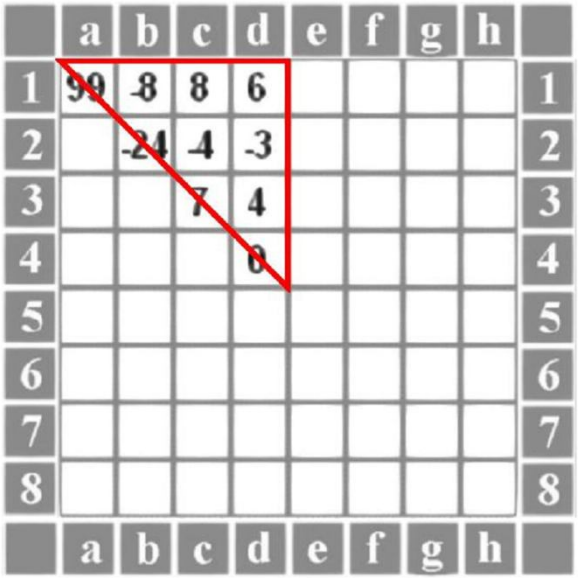


Fig. 11: Positional Strategy

2.4.1 MiniMax Algorithm

To find the best move during the game the player need to "look forward" a few moves. This is accomplished by searching in a game tree, in which the board positions are organized. **Fig. 12** shows a game search tree, wherein every node corresponds to a board position, the root of the tree is the starting position, or the position from which the player want to find a move.The children of each node N are the different positions that result from a single player's move inN. Thenodes without children in the game tree correspond to terminating positions.

Each node n is assigned a value to the board position by an analysis function A(n). Positive values of A(n) indicate that the position favors blue, while negative values mean that red is winning. Hence, for a position

n in which blue has won, maximum A(n) is ∞, while minimum A(n) is -∞ if red has won.
This value is computed by means of a position evaluation function and it indicates how good it would be for a player to reach that position. The player then makes the move that maximizes the minimum value of the position resulting from the opponent's possible following moves.

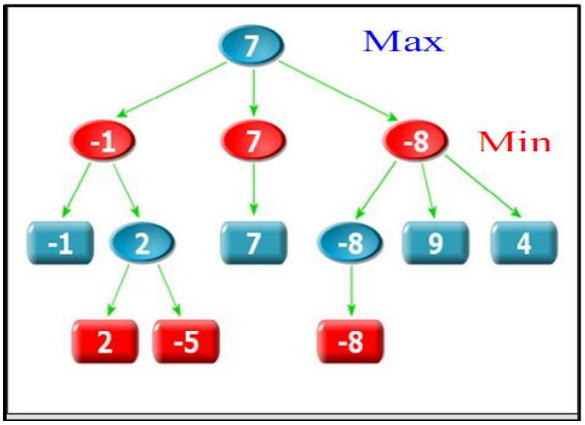


Fig. 12: MiniMax search tree

The analysis function is heuristic, it takes many different aspects of the board positioninto account, such as the number of discs, control of the board, certain key positions on the board, and mobility of player's disc.
For strategy games, where players take alternate moves, the MiniMaxalgorithm is widelyused, which is a simple, predictive game playing technique. Itis a decision rule for minimizing the possible loss for a worst case (maximum loss) situation. MiniMax explores all possible moves of the current player, simulating all valid moves, and exploring the possible moves of the opponent in response. It searches for the smallest score from the opponent's simulated responses, and the largest scores of its own simulated moves, thereby discovering which moves will be favorite for itsaction. This is a depth-first search method, which would take a long time to complete, so a depth limit will be employed to prevent the algorithm from taking too much time.
MiniMax strategy is part of the solution to a zero-sum game,it uses a recursive algorithm for choosing the next moveas shown in **Fig. 13**.

```
functionminimax(node, depth)
{
if(node is a terminal node or depth ==0)
return the heuristic value of node;
if(the adversary is to play at node)
{
let α := +∞;
foreach child of node
α := min(α, minimax(child, depth-1));
}
else          // we are to play at node
{
let α := -∞;
foreach child of node
```

Fig. 13: Pseudocode of MiniMax algorithm

```
architecture rtl of minmax is
begin
    process
    begin
        if ( pl_q ) then    -- me
            if ( ( max_p_q > best_value_q ) ) then
                best_value_d <= max_p_q;
                max_p_d <= max_p_q;
                if ( ( sp_q = 0  ) ) then
                    best_X_d <= last_X_q;
                    best_Y_d <= last_Y_q;
                end if;
            else    -- return best_value to upper level
                best_value_d <= best_value_q;
                max_p_d <= best_value_q;
            end if;
            if ( ( best_value_q > alfa_q ) ) then
                alfa_d <= best_value_q;
            end if;
            if ( ( best_value_q >= beta_q ) ) then
                sp_d <= ( sp_q - 1  ) ;
                pl_d <= (  not pl_q ) ;
                state_d <= EXPLORE_FETCH;
                fake_state_d <= EXPLORE_FETCH;
                max_p_d <= best_value_q;
            end if;
        else                -- oponent
            if ( ( max_p_q < best_value_q ) ) then
                best_value_d <= max_p_q;
                max_p_d <= max_p_q;
            else    -- return best_value to upper level
                best_value_d <= best_value_q;
                max_p_d <= best_value_q;
            end if;
            if ( ( best_value_q < beta_q ) ) then
                beta_d <= best_value_q;
            end if;
            if ( ( best_value_q <= alfa_q ) ) then
                sp_d <= ( sp_q - 1  ) ;
                pl_d <= (  not pl_q ) ;
                state_d <= EXPLORE_FETCH;
                fake_state_d <= EXPLORE_FETCH;
                max_p_d <= best_value_q;
            end if;
        end if;
        M_d(( ( Y_d * 8  )  + X_d ) ) <= '0';
    end process;
end;
```

Fig. 14 shows part of the MiniMaxalgorithmrealized in hardware using VHDL language.

III. RESULTS AND DISCUSSION

Fig. 15shows thiselectronic reversi gameusing an Altera DE2-70 FPGA development board.A embedded platform that contains an Altera Cyclone II 2C70 FPGAwith around 70K logic elements, wherein a soft core NIOS II can be implemented with ultimate flexibility in terms of reconfiguration ofactual hardware circuitry and intellectual property as well as on-board multimedia peripherals.

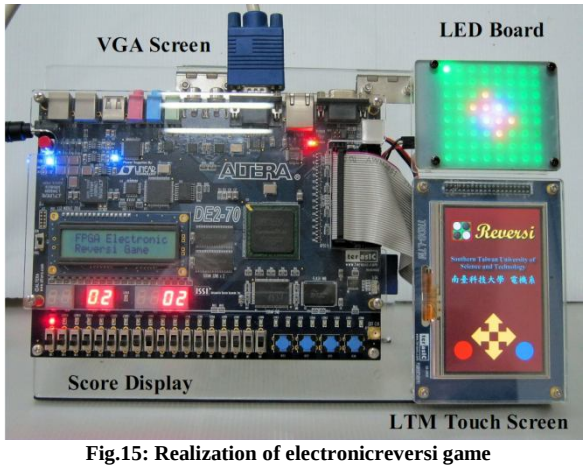


Fig.15: Realization of electronicreversi game

The game board will be displayed on the VGA screen (**Fig. 16)**, where the light green square indicates the disc to move, and the yellow squares are the legal place positions for player of blue discs.

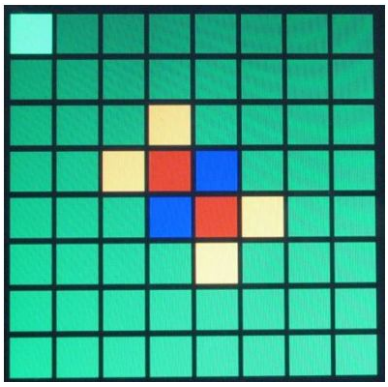


Fig.16: Reversi game board display on LCD screen

Tab.2 summarizes the result of 30 games by 3 players playing against the intelligent of the AI player implemented in this project with a search depth of 6.

Tab. 2: Results of game playing with computer

Game No.	Player	Game Results		
		Human (C)	AI (D)	Win Ratio (C/D)
1	**1**	**23**	**18**	**1.28**
2	1	33	41	0.80
3	1	11	24	0.46
4	**1**	**19**	**18**	**1.06**
5	1	10	19	0.53
6	1	12	28	0.43
7	1	14	37	0.38
8	**1**	**28**	**18**	**1.56**
9	1	4	45	0.09
10	1	11	45	0.24
11	2	9	43	0.21
12	**2**	**28**	**18**	**1.56**
13	**2**	**19**	**14**	**1.36**
14	2	24	39	0.62
15	2	11	53	0.21
16	2	20	44	0.45
17	2	21	43	0.49
18	2	22	41	0.54
19	2	17	25	0.68
20	2	11	53	0.21
21	3	13	50	0.26
22	3	8	56	0.14
23	3	0	59	0.00
24	3	4	58	0.07
25	3	12	52	0.23
26	3	6	56	0.11
27	3	3	53	0.06
28	3	17	34	0.50
29	3	16	41	0.39
30	3	2	62	0.03

Fig. 17 shows the resources usage of hardware in FPGA and memory requirement of software to realize this reversi game.

Flow Status	Successful - Sun May 03 09:10:18 2015
Quartus II Version	9.1 Build 222 10/21/2009 SJ Full Version
Revision Name	DE2_70_Reversi
Top-level Entity Name	DE2_70_Reversi
Family	Cyclone II
Device	EP2C70F896C6
Timing Models	Final
Met timing requirements	No
Total logic elements	9,525 / 68,416 (14 %)
Total combinational functions	8,812 / 68,416 (13 %)
Dedicated logic registers	3,644 / 68,416 (5 %)
Total registers	3660
Total pins	325 / 622 (52 %)
Total virtual pins	0
Total memory bits	672,592 / 1,152,000 (58 %)
Embedded Multiplier 9-bit elements	4 / 300 (1 %)
Total PLLs	1 / 4 (25 %)

```
Reversi_CPU --quartus_project_dir "C:/DE2_70_Reversi_0501" --jdi C:/DE2_70_Reversi
Info: (Reversi_Game.elf) 30 KBytes program size (code + initialized data).
Info:                    36 KBytes free for stack + heap.
```

Fig.17: Hardware and software resource usage

CONCLUSIONS

The electronic reversi game realized by this project demonstrates the advantageof hardware/software integration with minimum resources to achieve an intelligent system.Further improvements can be done. For examples,

1. The processing time of AI algorithm can be added for performance analysis in next version,
2. The MiniMax search depth of current version is fixed to 6, it should be adjustable to variable difficulty of game level in further version,
3. The DE2-70 board has equipped with Ethernet interface, therefore an internet connection can be expanded for remote game play.

REFERENCES

[1] "Reversi", Wikipedia, Feb. 2015. http://en.wikipedia.org/wiki/Reversi

[2] M.Buro,"LOGISTELLO", Mar. 2011. https://skatgame.net/mburo/log.html

[3] "Strategy Guide for Reversi& Reversed Reversi",Apr. 2011.http://www.samsoft.org.uk/reversi/strategy.htm

[4] J. L. Nielsen, "Reversi in Java", Sep. 2009.

[5] A.K. Peters, "Artificial Intelligence for Computer Games: An Introduction",Wellesley, MA. Jul. 2004. ISBN:1-56881-208-6.

[6] H. Ahmadi, "An Introduction to Game Tree Algorithms",https://sites.google.com/site/ctxtree/minimax

[7] J. Olivito, C. González and J. Resano, "FPGA Implementation of a Strong Reversi Player",International Conference onField-Programmable Technology (FPT), pp 501-507, Dec. 2010.ISBN: 978-1-4244-8980-0.

[8] "Nios II Processor Reference Handbook (Ver. 13.1.0)", Altera Corporation, Feb. 2014.

[9] "WS2812B Intelligent control LED integrated light source", Datasheet, Worldsemi Co.Limited, Jun. 2014.

[10] "LTM_User_Manual-4.3 Inch Digital Touch Panel Development Kit",Terasic Inc., Jul. 2011.

★★★