A Case Study of Connect6 Game FPGA-based Implementation Using the Multi-turn Prediction Algorithm

Trong-Thuc Hoang*, Vi-Thuy Tran, Thanh Le, Minh-Triet Luu
Cao-Quyen Tran, Xuan-Thuan Nguyen†, and Trong-Tu Bui‡
Faculty of Electronics and Telecommunications
The University of Science, Ho Chi Minh City
227 Nguyen Van Cu St., Dist.5, Ho Chi Minh City, Vietnam
*†‡{htthuc, nxthuan, bttu}@fetel.hcmus.edu.vn

Abstract*— Designing an intelligent system that can play chess against human is challenging. Such a system assists in analyzing chess and researching into human cognition. This paper presents a case study of Connect6 game hardware design. The Connect6 game, a kind of 6-in-a-row game, is first introduced by Professor I-Chen Wu, National Chiao Tung University, Taiwan, in 2003. There still have no effecient algorithms with their hardware implementations to forecast many turns in advance, though many have been proposed to enhance the prospect of the victory. For this reason, the multi-turn prediction algorithm, which can detect almost all of the situations and deploy the successful strategies, is presented in the paper. Our design is implemented on an Altera DE2 board with the Cyclone II FPGA chip and the operation clock frequency of 20 MHz. The result shows that the proposed design achieves the winning percentage of 92% within the time limit of 5ms for each of its turn.

Keywords*— case study, Connect6, FPGA, hardware, multi-turn prediction.

I. Introduction

"The Connect6 is a fair and highly complex game with simple rules" [1]. Two players, with dark and light stones, alternately place two stones on empty intersections of a 19x19 GO board for a turn. The initial turn is special because of the dark side always plays first and put one stone in the initial turn. The winner is the one who gets six stones in-a-row first (i.e. horizontally, vertically or diagonally). Since 2003, many Connect6 tournaments have been organized around the world [2]-[4].

Although the rule is simple, a computer cannot solve all possible situations due to a number of intensive computational processes. For this reason, many different Connect6 solutions are implemented in software and/or hardware. Junjie Tao et al. [5] proposed a method to construct an opening book that "contains an ocean of grandmasters game records or the excellent positions produced between computers and grandmasters", while I-Chen Wu et al. [6] presented a grid computing environment for Connect6 applications. However, those methods consume high cost and high power due to the powerful computers. To avoid utilizing computers, Kentaro Sano et al. [7] designed a hardware and software co-design of a Connect6 AI with scalable streaming cores in an FPGA. Although the latency was improved, it was too far to reach the time limit of almost Connect6 contests (i.e. 1 second for each turn). Takahio Watanabe et al. [8] presented an approach of hardware accelerator implementation in an FPGA using two-stage pipelined evaluation. However, the logic utilization is too large to implement in such a low-cost FPGA prototype. In addition, the others experimental results and analysis of the systems are not discussed.

It is the fact that an effective Connect6 strategy must balance the offensive moves with the defensive ones. The offensive strategy looks for the winning moves or at least increases the chance by each stone be placed. The defensive strategy prevents the opponent from winning or creating a serious threat that could make a winning sequence. In other words, a combination of the offensive and defensive strategies will lead to a successful Connect6 algorithm.

In this paper, we present a multi-turn prediction algorithm for the FPGA-based implementation. This algorithm consists of seven steps that can predict the three turns in advance. Furthermore, our proposed Point-Quality technique which is used among the steps assists the FPGA in making the best move decision. The flow of the design is as follows. Firstly, Verilog HDL and Modelsim are utilized to develop and simulate the Connect6 hardware, respectively. After successful simulation, Altera Quartus II and SignalTap tool are employed for the synthesis and verification, respectively. The target platform is an Altera DE2 development kit [9] equipped with a Cyclone II EP35F672C6 FPGA chip. Lastly, the completed system plays against the opponents so as to vertify the effecitiveness of the algorithm and the implementation. The opponent can be a Connect6 contest program [2], another FPGA system, or an experienced human player.

The remainder of this paper is organized as follows. Section 2 presents the notations that are used in the paper with their examples. The details of the strategy game and the algorithm are discussed in Section 3. The hardware architectures is described in Section 4. The experimental results that portray the effectiveness of the FPGA implementation are presented in Section 5. Finally, the conclusion is given in Section 6.

978-1-4673-5604-6/12/$31.00

Table 1. Summary of the notations that are used in the paper.

Notation	Meaning	Defined in
L6	Six same-color stones are lined up	Fig. 1(a)
L5	Five same-color stones and one empty cell are lined up	Fig. 1(b)
L4	Four same-color stones and two empty cells are lined up	Fig. 1(c)
L3	Three same-color stones and three empty cells are lined up	Fig. 1(d)
L2	Two same-color stones and four empty cells are lined up	Fig. 1(e)
L1	One same-color stone and five empty cells are lined up	Fig. 1(f)
L3_P1	A type of L3 which becomes 1TW_T1 by adding one stone	Fig. 1(g)
L3_P2	A type of L3 which becomes 1TW_T2 by adding one stone	Fig. 1(h)
L2_P1	A type of L2 which becomes 1TW_T1 by adding two stones	Fig. 1(i)
L2_P2	A type of L2 which becomes 1TW_T2 by adding two stones	Fig. 1(j)
1TW	One-turn win	Sec. II-A
1TW_T2	One-turn win with two threats	Sec. II-A
1TW_T1	One-turn win with one threat	Sec. II-A
2TW	Two-turn win	Sec. II-B
3TW	Three-turn win	Sec. II-C
3TW_C	Three-turn chain win	Sec. II-C
3TW_NC	Three-turn not-chain win	Sec. II-C
PQ	Point-Quality Technique	Sec. II-D
GI	Gain-Initiative	Sec. III
MPC	Most Potential Cell(s)	Sec. III

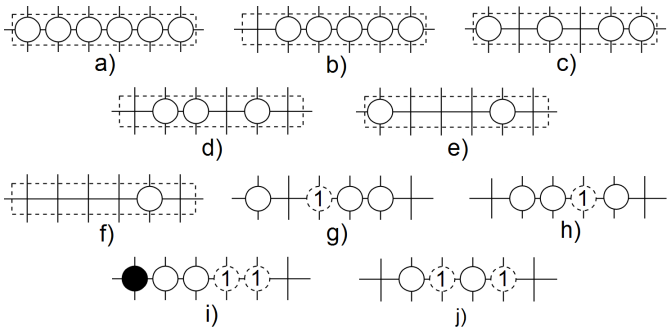

Fig. 1. Examples of L6 (a), L5 (b), L4 (c), L3 (d), L2 (e), L1 (f), L3_P1 (g), L3_P2 (h), , L2_P1 (i), and L2_P2 (j) for the light side.

II. Notations

The notations in this paper are summarized in Tab. 1 and will be used in the following sections to develop an unifying framework for our new algorithm.

A. One-turn Win

The 1TWs are all of the situations that lead to win the game within one turn. Winner is the one who had L6. Before having L6, winner has to obtain the L4 or L5. Hence, the 1TWs denote all of L4s and L5s situations.

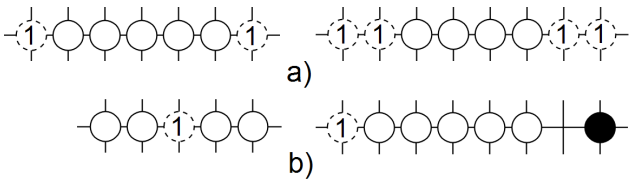

Fig. 2. Examples of 1TW_T2 (a) and 1TW_T1 (b) for the light side.

A threat is defined as a position which needs to be placed a stone in order to prevent the FPGA (or the opponent) from winning. The 1TW_T1 and 1TW_T2, two types of the 1TW, have one and two threats, respectively, as portrayed in Fig. 2.

B. Two-turn Win

The 2TWs are all of the situations that lead to win the game within two turns. If the FPGA creates more than two threats in a turn, its victory will be secured by the next turn. Therefore, the 2TW can be defined as a way that creates two 1TWs after a turn, and at least one of them is a 1TW_T2. There are five types of 2TWs, as illustrated in Fig. 3.

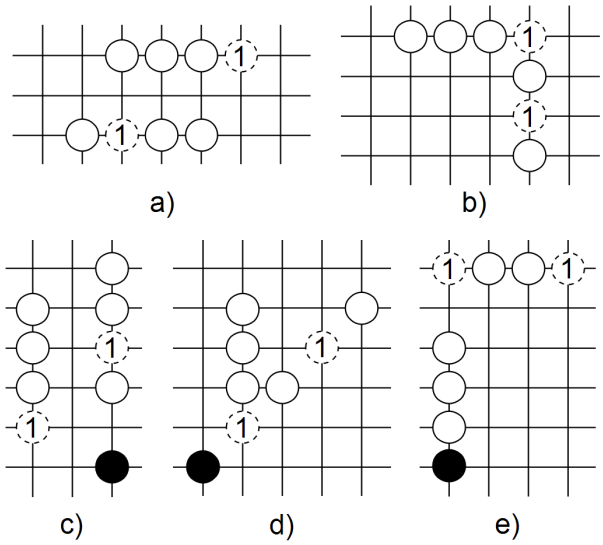

Fig. 3. Examples of five types of 2TWs for the light side: two L3_P2s (a), L3_P2 intersects L2_P2 (b), L3_P2 and L3_P1 (c), L3_P2 intersects L2_P1 (d), and L3_P1 intersects L2_P2 (e).

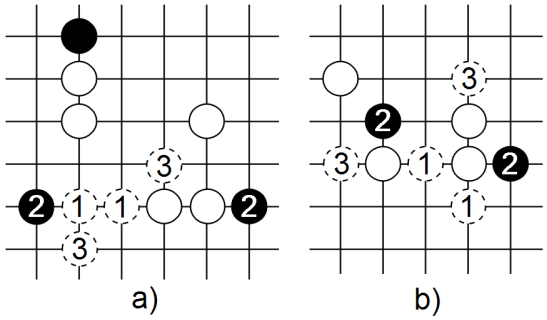

Fig. 4. Examples of 3TW_C (a) and 3TW_NC (b) for the light side.

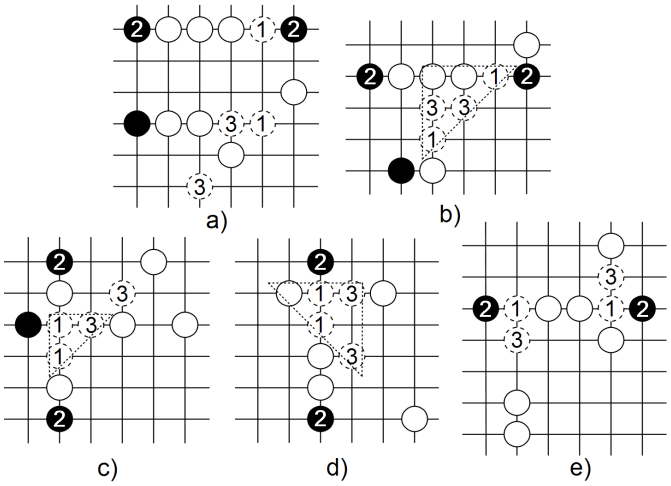

Fig. 5. Examples of five groups of 3TW_Cs for the light side: L3_P2 and L2 intersect L2 (a), a triangle made from L3_P2 - L2 - L1 (b), a triangle made from L2_P2 - L2 - L1 (c), a triangle made from L2_P2 - L2 - L2 (d), and an H-formation formed by L2_P2 with two L2s (e).

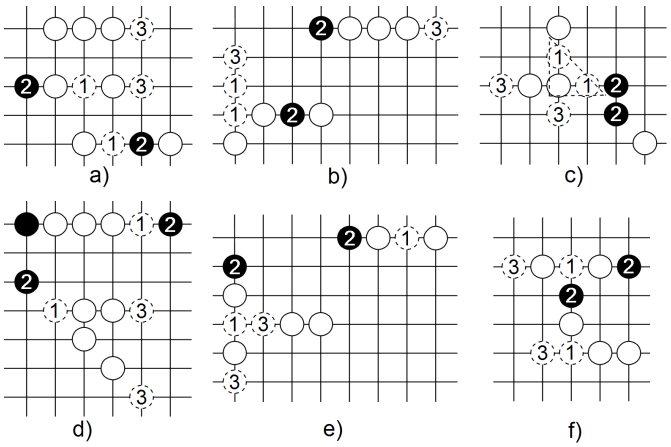

Fig. 6. Examples of six groups of 3TW_NCs for the light side: L3_P2 with two L2s (a), L3_P2 and L2 intersects L1 (b), a triangle made from L2_P2 - L2 - L1 (c), L3_P1 and L2_P2 intersects L2_P2 (d), one L2 and two L2s intersects with each other (e), and H-formation formed by L2 - L1 - L2 (f).

C. Three-turn Win

The 3TWs are all of the situations that lead to win the game within three turns. The 3TWs are divided into two types: 3TW_Cs and 3TW_NCs, as illustrated in Fig. 4(a) and Fig. 4(b), respectively. After a turn, a 3TW_C can bring about one 1TW and one 2TW while a 3TW_NC creates two 2TWs. 3TW_C and 3TW_NC situations can be classified into groups, as shown in Fig. 5 and Fig. 6, respectively.

D. The Point-Quality Technique

The Point-Quality (PQ) of a cell denotes the total qualities of four lines (i.e. horizontally, vertically or diagonally) that cross over that cell. The PQ is used for an empty cell only and calculated by the following equations.

$$PQ = LQP + DPQ \quad (1)$$

$$LPQ = LPQ_H + LPQ_V + LPQ_{D1} + LPQ_{D2} \quad (2)$$

$$DPQ = DPQ_H + DPQ_V + DPQ_{D1} + DPQ_{D2} \quad (3)$$

where, LPQ and DPQ are the total qualities of light lines and dark lines, respectively. There are four light lines (i.e. LPQ_H, LPQ_V, LPQ_{D1}, and LPQ_{D2}) which correspond to four directions: Horizontal (H), Vertical (V), Diagonal 1 (D1, from top-left to bottom-right), and Diagonal 2 (D2, from bottom-left to top-right), respectively. Similarly, four dark lines in Equation (3) correspond to the four directions, too.

The quality computations of all lines are similar despite the sides and directions differences. The LPQ_H computation is taken for example in Fig. 7. The example computations are for the empty cell A which is in the center of each figure. Fig. 7(a) shows the points of the cells which stand near the center cell A. It is clear that the closer cell has the higher quality than the futher one as can be seen in Fig. 7(a). Because of the computation is for the light side, a dark stone on this line will be considered as a blocking stone. When a blocking stone appears, the qualities of the cells between the blocking stone and the center cell A are decreased. Fig. 7(b) shows cell points with blocking stones. Noticed that if the length of a blocked line (i.e. between two blocking stones) is smaller than six cells, that line will become useless and all cell points are zero. Several examples of the LPQ_H computation are depicted in Fig.7(c). For example, the LPQ_H value of the figure that has two light stones and no blocking stone is equal to 9 because of the two light stones are placed on the 5-point and 4-point cells. The last figure in Fig. 7(c) has the LPQ_H value is 0 because the length of the blocked line is smaller than six.

In addition, Fig. 8(a) shows an example of using PQ for blocking the dark side 1TW_T1. In the example, the light side has two options to places a blocking stone. The best option is chosen by selecting the cell which has the largest PQ.

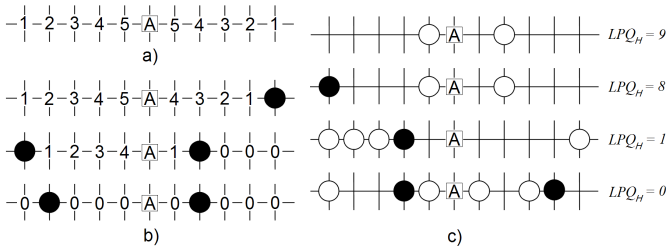

Fig. 7. Cells' points with no blocking stone (a); Cells' points with blocking stones (b); Examples of LPQ_H computation (c).

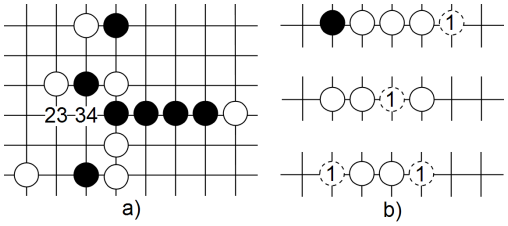

Fig. 8. Example of using PQ to make decision (a); Examples of GI (b).

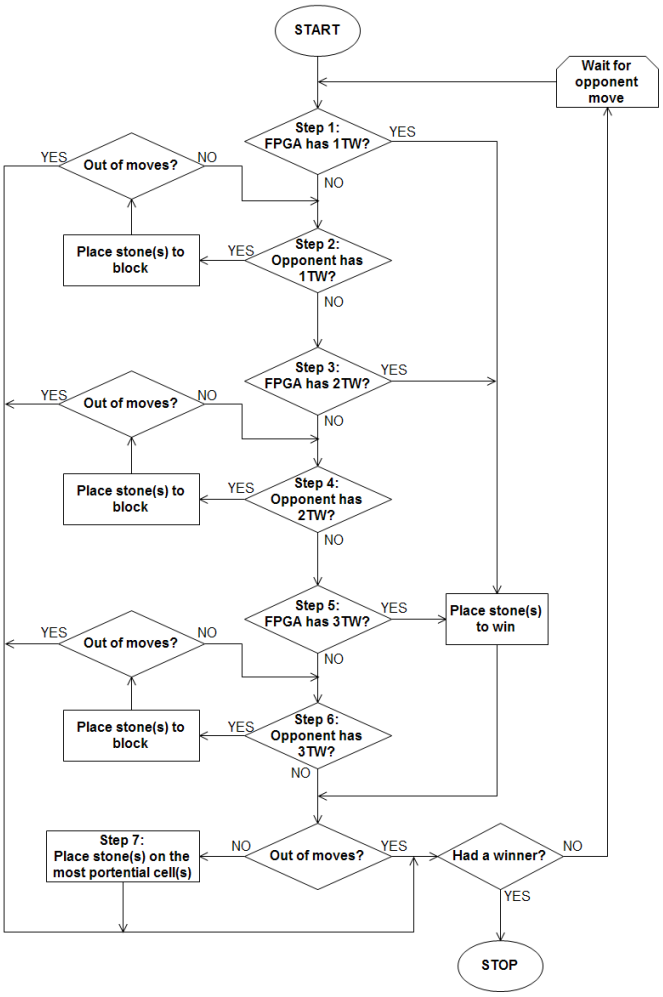

Fig. 9. The flowchart of Connect6 game strategy.

Fig. 10. The re-organization of the seven steps.

III. Algorithm

The algorithm is based on the winning moves prediction of both the FPGA and opponent. The game strategy is divided into seven steps with different priorities as depicted in Fig. 9. Each step is derived from both game strategies and match experiments so that almost of the winning moves can be forecasted.

First of all, the FPGA finds a way to win the game by searching its 1TW in the Step 1. If the FPGA does not have any 1TW, it finds and blocks block the 1TWs of the opponent in the Step 3. Similarly, the searching for the 2TWs of both the FPGA and the opponent is done by the Step 3 and the Step 4, respectively. In the same manner, the searching for the 3TWs of both sides is done by the Step 5 and the Step 6. After the Step 6, if the FPGA still has moves decisions to make, the final step, Step 7, will look for the best empty cells to place a stone or two in order to complete the turn. Otherwise, the FPGA skips the final step. If the FPGA successes in the offensive step (i.e. Step 1, Step 3, and Step 5), the remaining steps after the succeed step will be skipped. In contrast, the FPGA has to repeat the defensive steps (i.e. Step 2, Step 4, and Step 6) after it succeed in blocking the opponent's winning moves. Because the FPGA has to make sure that the opponent does not have any winning move before it proceeds to the next step.

As mentioned above, a 3TW_C creates one 1TW and one 2TW after a turn. If the FPGA has a 3TW_C, the opponent's 2TWs will be out of concern. Because after creating one 1TW and one 2TW, the FPGA forces the opponent to block the 1TW, then the FPGA victory will be ensured by the remaining winning moves - the 2TW. Therefore, the FPGA has to find its 3TW_C to win before find the opponent's 2TWs to block. However, the FPGA can only find its 3TW_NC only when the opponent does not have any 2TW. As a result, the Step 5 is divided into two sub-steps. There are Step 5A and Step 5B for seeking the 3TW_Cs and 3TW_NCs, respectively. The Step 5A is swapped with the Step 4 because the above reason. Similarly, the Step 6 is divided into two sub-steps: Step 6A and Step 6B. With the same reason, the Step 6A is put before the Step 5B. To sum up, Fig. 10 describes the re-organization of the seven steps in order to build the best game strategy.

The final step is divided into two sub-steps: the GI step and the MPC step. Understanding the importance of the initiative, the GI step creates the 1TWs by looking for the uncompleted L4s (i.e. L3_P2, L2_P2, L3_P1, L2_P1) and filling them into a completed one. The GI step examples are shown in Fig. 8(b). If the FPGA still has moves after the GI step, the MPC step will make the move decisions. The MPC step uses the PQ technique to estimate the potential of all empty cells on the board. Then, the MPC step places a stone on the cell which has the largest PQ. The MPC step repeats its processing until out of moves.

IV. Proposed Hardware Architecture

Fig. 11 describes a match between the FPGA and the opponent. The opponent can be a software, another FPGA, or a human player. The block diagram of the system is shown in Fig. 11(a). Beside the Connect6 Core (CC), a 2-Kbit On-chip Memory (MEM), an UART Controller (UC), and a Control Unit (CU) are designed to store the 19x19 GO board, to communicate with the opponent, and to control the whole operation of the system, respectively. The positions of all stones on the GO board are stored in the MEM. There are 10-bit MEM address, which is formed by 5-bit row and 5-bit column, and 2-bit MEM data to indicate the position and the state (i.e. empty cell, filled with dark stone, filled with light stone) of each intersection on the GO board, respectively.

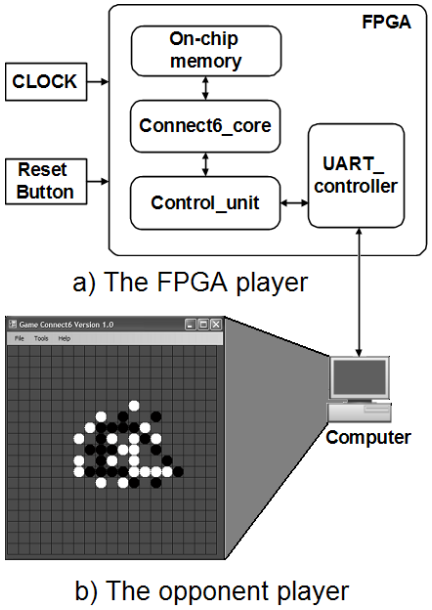

Fig. 11. The block diagram of the Connect6 system on FPGA chip (a) and The PC software interface (b).

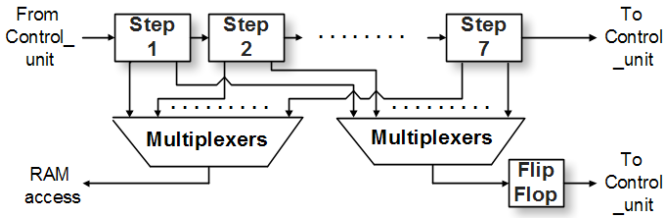

Fig. 12. The Connect6 Core (CC) design.

After start-up, the system waits for the opponent choice (i.e. dark side or light side). The choice is transferred from the PC to the CU by the UC as ASCII string. If the FPGA plays first, the CU will place the initial stone at the $(10, 10)$ position. Otherwise, the CU receives the initial stone from the PC. In the normal turn, the CC makes a move decision based on the multi-turn prediction algorithm. The move positions are transferred from the CC to the CU as 10-bit MEM address for each move. Subsequently, the CU converts 10-bit MEM addresses to ASCII strings, then it transmits the strings to the PC by the UC. In an oppoent's turn, the CU receives the move positions from the PC as ASCII strings. It, then, converts the strings to 10-bit MEM addresses and writes them to the MEM.

The CC accesses the MEM during its processing in order to find the winning moves. After the CC is done, it transmits the result to the CU. Therefore, the CC's ports are connected to the CU and the MEM as can be seen in Fig. 12. The CC module is divided into seven sub modules, each sub module implements one of the seven steps in the algorithm. A sub module requires four signals to communicate with each other: iRst_n, iTurn, oDone, and oTurn as depicted in Fig. 13. The next step is allowed to operate only when the previous step finished. This is done by connecting the oDone[i-1] to iRst_n[i], where i = 2, 3 7. Also, iTurn[i] and oTurn[i] give the number of the remaining moves before and after the step[i]. The Step 1 has iTurn[1] = 2 as default, as a result of which the Step 1 does not have iTurn signal. The operation of the CC is completed upon the oDone[7] is asserted. Because of only one step is activated at a time, the multiplexers are needed to switch the MEM and flip-flops access.

Because the purposes of the steps are different, their designs are different, either. However, they share the similar method that will find a line (i.e. L5, L4, L3, L2) and check the winning moves (i.e. 1TW, 2TW, 3TW) based on that line. Although steps designs are different, they have the similar hardware construction as shown in Fig. 14. The MEM addresses are accumulated by the SCAN and the MEM data are transferred to the DETECTOR in order to search for a line. When a line is found, the STEP_FSM processes further on that line to detect the winning moves. Therefore, the STEP_FSM can decide where to put the stones. The support modules (i.e. H_module and Triangle_module) may be required in some steps to support the STEP_FSM processing. A step processing is done whenever the SCAN is done, out of moves, or the step found a winning moves if it is an offensive step.

V. Experimental Results

The proposed hardware architecture is implemented on an Altera DE2 board with the Cyclone II EP2C35F672C6 FPGA chip. In Fig. 15, a completed system in the DE2 board is playing against the opponent by COM port. Tab. 3 gives the worst execution latency in each step. The worst case of a step occurs when that step did not make any move. The total latency is about 5.3 ms at the clock frequency of 20 MHz, which is nearly 350 times higher than the software, FPT2011 AI-GUI [2] which is operated on an Intel Core i3 CPU-based PC. Table 2 compares the logic utilization and operating frequency of our implementation with the others.

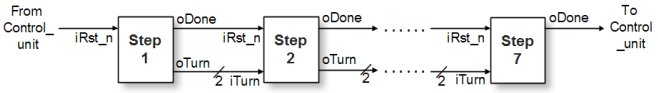

Fig. 13. Steps communication.

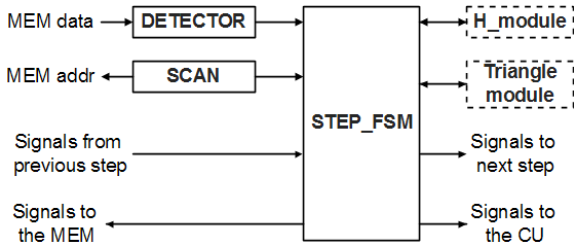

Fig. 14. A general step module design.

Table 2. The comparison of the proposed Connect6 implementation with the others.

RESOURCE	OUR DESIGN	DESIGN [8]	DESIGN [10]	DESIGN [11]
ALUTs	23,420 (71%)	104,919 (92%)	1,615 (NA%)	NA
Registers	8,446 (25%)	7,884 (6.9%)	3,329 (NA%)	NA
Memory Bits	2,048 (0.45%)	219x9 Kbit (50.7%)	NA	0 (0%)
Slices	NA	NA	2,211 (37%)	5,184 (37%)
Frequency	20 MHz	20 MHz	50 MHz	100 MHz
Prototype	Altera Cyclone II EP2C35	Altera Cyclone IV EP4CE115	Xilinx Spartan 3 700A-4	Xilinx Virtex II XC2VP30

Table 3. The execution time (us) for each step.

Step	Time(μs)	Step	Time(μs)	Step	Time(μs)
1	77.95	5A	155.9	5B	155.9
2	77.95	4	155.9	6B	77.95
3	155.9	6A	77.95	7	4,275.9

In addition, to evaluate the performance of the hardware implementation, 100 game matches are organized for 25 skillful volunteers. Each volunteer played against the FPGA 4 times, two times with dark side and two times with light side. The FPGA won 92 matches within the maximum of 58 turns. Those results are showed in Fig. 16, where Y-axis is the number of turns in a match and X-axis is the number of matches which was finished after a number of turns. For example, according to the first and the last column, three matches and one match were finished after eight turns and fifty eight turns, respectively.

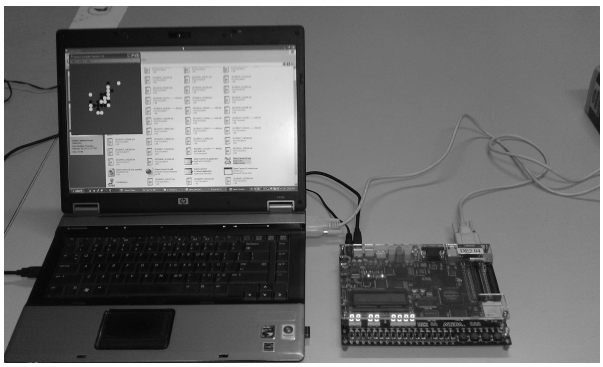

Fig. 15. The completed system is playing against the opponent.

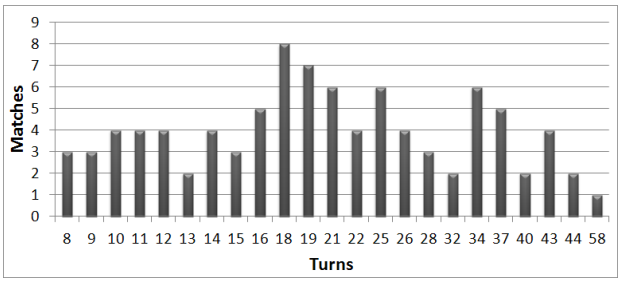

Fig. 16. The matches summary between the FPGA system and human players.

VI. Conclusion

We have presented a multi-turn prediction algorithm for an FPGA-based Connect6 implementation. By using seven steps employing a proposed Point-Quality technique, the FPGA can predict three turns in advance and give the best moves decisions. Although the logic utilization is suitable for most low-cost FPGA chips, the winning percentage is very high, up to 92%, and the execution time is quite low, about 5ms, in the comparison with the other implementations. The algorithm could be improved further by adding a four-turn win prediction together with some hardware design techniques. Therefore, the FPGA can forecast almost all of the winning moves to increase its percentage of winning, and maintains the time limit within 1 second for each turn.

References

[1] (2007, Oct.) Connect6 Homepage. [Online]. Available: http://www.connect6.org/

[2] (2012, Jan.) FPT 2011 Design Competition. [Online]. Available: http://www.eecg.toronto.edu/~janders/FPT_2011_competition/

[3] (2012, Jan.) Connect6 tournament at the 11th Computer Olympiad. [Online]. Available: http://connect6.csie.nctu.edu.tw/

[4] (2012, Jan.) The First Annual NCTU Cup Connect6 Open Tournament 2006. [Online]. Available: http://www.connect6.org/contest.html

[5] Jun-jie Tao, Chang-ming Xu, Kang Han, and Xin-he Xu, "Construction of Opening Book in Connect6 with Its Application," in *Proc. IEEE Int. Conf. Control and Decision*, Guilin, China, June 2009, pp. 4530-4534.

[6] I-Chen Wu, Chingping Chen, Ping-Hung Lin, Guo-Chan Huang, Lung-Ping Chen, Der-Johng Sun, Yi-Chih Chan, Hsin-Yun Tsou, "A Volunteer-Computing-Based Grid Environment for Connect6 Applications," in *Proc. IEEE Int. Conf. Computational Science and Engineering*, Vancouver, Canada, Aug. 2009, pp. 110-117.

[7] Kentaro Sano, "SW and HW Co-design of Connect6 Accelerator with Scalable Streaming Cores," in *Proc. Int. Conf. Field-Programmable Technology (FPT)*, New Delhi, India, Dec. 2011, pp. 1-4.

[8] Takahio Watanabe, Retsu Moriwaki, Yuichiro Yamaji, Yuki Kamikubo, Yuki Torigai, Yuki Nihira, Takashi Yoza, Yumiko Ueno, Yuji Aoyama, Minoru Watanabe, "An FPGA Connect6 Solver with a Two-Stage Pipelined Evaluation," in *Proc. Int. Conf. Field-Programmable Technology (FPT)*, New Delhi, India, Dec. 2011, pp. 1-4.

[9] (2011, Dec.) Altera DE2 Development and Education Board. [Online]. Available: http://www.altera.com/education/univ/materials/boards/de2/unv-de2-board.html

[10] Kizheppatt Vipin, Suhaib A. Fahmy, "A Threat-Based Connect6 Implementation on FPGA," in *Proc. Int. Conf. Field-Programmable Technology (FPT)*, New Delhi, India, Dec. 2011, pp. 1-4.

[11] Ziermann, T., Schmidt, B., Muhlenthaler, M., Ziener, D., Angermeier, J., Teich, J., "An FPGA implementation of a threat-based strategy for Connect6," in *Proc. Int. Conf. Field-Programmable Technology (FPT)*, New Delhi, India, Dec. 2011, pp. 1-4.