Eng. & Tech. Journal, Vol.30, No.7, 2012

An FPGA Based a Digital Circuit Design for Route Optimization

Ivan A. Hashim
Electrical Engineering Department, University of Technology/ Baghdad
Email:k300@yahoo.com

Received on: 2/2/2011 & Accepted on: 2/2/2012

ABSTRACT

Route optimization is searching problem to find the shortest path from starting to end point within certain criteria. In this paper, a digital circuit design implementation was presented according to the Dijkstra algorithms and with new digital technology. The proposed circuit is built using VHDL and simulated using Xilinx ISE 9.2i package. The test of the implemented circuit was made by use a 25 point network map to select the shortest path between any two specific points (from point 3 to point 24). Simulation behavioral model results show that proposed circuit satisfies the specified operational requirements. The result appears this requirement with a short time (depend on the clock frequency used 50MHz). Furthermore, this circuit is flexible to increase the number of point in the map network.

Keywords: VHDL design, Route Optimization, Searching problem.

اعتماد مصفوفات البوابات المنطقية القابلة للبرمجة في الميدان في تصميم الدائرة المنطقية الخاصة بالطريق الأمثل

الخلاصة

الممر الأمثل هي مشكلة البحث لإيجاد الممر الأقصر من نقطة البداية إلى نقطة نهاية (هدف). هذا البحث يقدم تصميم لدائرة الكترونية رقمية وفق نظام ديجكسترا وباستخدام تقنية حديثة رقمية قابلة للبرمجة . تم بناء الدائرة المقترح باستخدام VHDL وتمت محاكاته باستخدام Xilinx ISE 9.2i package. لفحص هذه الدائرة تم استخدام نموذج مؤلف من 25 نقطة مرتبة لاختيار الممر الأقصر بين أي نقطتين معينتين (من نقطة 3 إلى نقطة 24) . لقد أظهرت نتائج محاكة هذا النموذج بان الدائرة المقترح قد حقق المتطلبات التشغيلية المحددة. وكانت النتائج تظهر الغرض من التصميم. الوقت المستغرق يعتمد على تردد النبضة(50 MHz) ويمكن تقليل الوقت بزيادة تردده. كذلك فان هذه الدائرة مرنة عند زيادة عدد النقاط في المخطط.

https://doi.org/10.30684/etj.30.7.2
2412-0758/University of Technology-Iraq, Baghdad, Iraq

INTRODUCTION

Routing problems are searching problems for finding an optimal route from an origin (source) to a destination on a map network within a time limit. In a practical system, the time means cost. There are two important applications of this subject; rout optimization for computer network and rout optimization for car network. In these applications, the optimum means low cost from beginning to the end with continuous evaluation of cost until reach to the destination [1].

In the first application, the information data must reached to the destination point within less time and lower cost. As well as in the second application, it will represent the user guide to reach to the destination point easily and quickly with low cost. The low cast has a wide meaning, for the first application its explained in [2], and for the second application mean less time, less fuel, less emission (spatially CO_2).

There are several researchers discussed the rout optimization, which are:
M.R.Delavar, 2000, [1], proposed a solution by uses a genetic algorithm (GA), where a part of an arterial road is regarded as a virus. Then, he generates a population of viruses in addition to a population of routes. A customized method based on a genetic algorithm has been proposed and successfully implemented in a certain area by using the optimal combination of viruses. MuktiAdvani 2003, [6], his case study of Bhavnagar district area was conducted for determining the optimal routes from one origin to many destination kinds of the problem, with an objective of minimizing travel distance and travel time of users. It constrains taken into consideration where impedance for intersections, type of road and speed by use GIS (Geographical Information System).

S. A. Zargar 2006, [7] he researched by using the generic algorithms. Wad presented a model of decision making in a considered one way – two way streets are developed. The efficiency of the model in Qazvin network is shown and the results compared with the current situation as a case study. The objective function of the research is to minimize the total travel time for all users, which is one of the most used in urban network objectives.

M. Saffarzadeh, 2007 [8], was presented two mathematical models for alignment in level areas. In the first model, the forbidden zones are not considered, but in the second model the forbidden zones are prevented from being passed through by modeling these zones in a circular shape. In this research, the mathematical non-linear programming has been used for modeling.

P. Shrivatava, 2007, [5], the objective of his proposed research is a development of a feeder bus route network and coordinated schedules for public buses for the existing schedules of main transit (suburban train) at the given Dublin (Ireland) Area Rapid Transit (DART) station. To satisfy the entire demand, the routes developed by GAs were modified by a well-designed heuristic approach.

A. Nagar, 2007. [4], his research aims to investigate the application of computer modeling and graphical simulation techniques for supporting the evaluation of routes

in road networks from a range of design and user related perspectives, and present a multi-criteria based approach to prototyping road networks.

Tien Vu, 2009, [9] his invention relates to a method and apparatus for a navigation system to determine a route to a destination, and more particularly, to a method and apparatus for a navigation system for establishing the most preferable calculated route to a destination when such a destination is an intersection by attaching a dummy link of zero-length to the intersection destination.

P. G. Hartman, 2010, [10], he present a method of planning a vehicle route includes estimating fuel requirements of identified alternative routes or route segments using navigation vehicle, and power train information.

The goal of this paper is to present a design of a Digital Circuit for Route Optimization based on FPGA. This circuit has found the shortest route between two specific points according to Dijkstra's algorithm. Furthermore, the proposed circuit can neglect the unavailable path (blocked) when this path is closed for any reason then reconfigure the shortest route again depending on the available paths.

OPTIMUM PATH SELECTION USING DIJKSTRA'S ALGORITHM

In most time the people choose the lowest-cost path on the route network map when traveling from a point (node) to any other one. There are many paths connecting node to node; however, focusing solely on the path with a minimum cost to connect two nodes. Under this condition, the cost between nodes can be calculated by Dijkstra method [3]. Dijkstra's algorithm is used to choose the optimum path, i.e. the path with minimum cost. In addition, it's solved the single source shortest path problem on a weighted, directed graph in which all edge weights are non-negative. The cost function is designed with the objective of determining the cost of the road according to the multi objective criteria. The cost function is applied to all possible roads to reach the destination point from a starting point [4]. The selection of the optimum path is based on the cost of the path, denoted as C_f.

$$C_f = \sum_{i=1}^{n} C_i * W_i \qquad ...(1)$$

Where;

n: is the number of criteria (safety, comfort, view).

C_i : is the cost of the road for criterion.

i, W_i : is the weight of criterion i provided by the user.

There is an important factor which is time traffic surveys (also can take it with a cost) were conducted during the morning peak period (i.e., 7–9AM). Since the maximum number of commuters' travels from 8–9AM, this time period is identified as the peak hour. It was confirmed during the surveys that after the 9AM commuter traffic start decreasing and becomes much less after 9.30AM [5]. Moreover, from 2-3 Pm is a max, but at the reverse direction.

PROPOSED WORK

The hardware implementation of the proposed digital circuit has been performed using the Xilinx technology [Xilinx Spartan-3 XC3S200]. The proposed circuit has the ability to select a shortest path between any two selected points in the network node distribution shown in Figure (1). The flowchart of the proposed work shown in Figure (2) has been built according to Dijstra's algorithm. Figure (3), represents the VHDL program of the route circuit, where the selection process and its statements have been demonstrated. There are 25 points arranged as five rows and five columns have been implemented for test purposes, as shown in Figure(1), each point , in certain column ,has been connected to all points in the next column.

Each path has certain values, which represents the cost of it. According to these assumptions, Table (1) represents the values of paths and blocks. The block values represent the availability of the related path, when the block value of a certain path equal 1 means the path is available. The block values were used in the implementation as enable value to take a certain path into account or not. The data which will be entered to the circuit are two types. The first type is from the user (NSD) which represents the source and destination points. The other type of data is from the management centers (which collect the data about the paths from many data sources). Moreover, this data is divided into two kinds; one of them is the path data (PFT) which means the path data from the point to point and the other is a block data (BFT) which means the block connection data for any two points. Therefore, at the initial time of the work the values of the network map will be interred according to nodes distribution. Then, these values will be continuously coming and entered from the management center at all time. Accordingly, the evaluation will be continuously and the reevaluation will be take place every time as long as the data reached to find the new shortest path according to it. According to this description the results of the route circuit were made for two cases as shown in Figure (4). For first case, the network node distribution map, shown in Figure (1), has been implemented according to the values of Table (1). In this case, it has been found that the shortest path is [3-10-13-19-24 with total cost 12], as shown in Figs (4a-4c).

Path 1: 3 — 7 — 11 — 17 — 24 = path cost 29

Path cost value from point 17 to point 24

Path 2: 3 — 8 — 12 — 17 — 24 = path cost 19

Path 3: 3 — 10 — 13 — 19 — 24 = path cost 12 shortest path is (3)

In the second case, changes have been made in both path and block values, according to Table (1). In this case, the shortest path has been reconfigured to [3-7-13-17-24 with path cost is 13]. As shown in Figs (4d-4e).

Path 1: 3 6 12 17 24 = path cost 26

Path 2: 3 7 13 17 24 = path cost 13 shortest path is (2)

Path 3: 3 8 12 17 24 = path cost 19

Path 4: 3 10 13 17 24 = path cost 14

Path 5: 3 10 13 19 24 = path cost 20.

In Figure (1), the shortest path was indicated by green color, other available paths are indicated by red color and the un-reached or cutting path by black color while the other blocked paths by dashed lines.

DISCUSSION

As state in the proposed work and as shown in the Figures (4-a, d, and c) the input data are;

PFT[18:0]--- 19-bit :represent the first type of data entered to the circuit, its bits are distributed as [5-bit for no. of each point, 8-bit for the value of the path cost and one-bit for enable].

BFT[11:0]---12-bit: represent the second type of data entered to the circuit, its bits are distributed as [5- bit for no. of each point, one-bit for block value and one-bit for enable].

NSD[10:0]---11-bit also, 5-bit for no. of each point, one-bit for enable.

The enable data was used to take the values from buffer make a reevaluation only when a new data reach.

As shown in the Figures (4a-4e) the initial input data are completely entered serially. The serial input data will reduce the number of FPGA IC pins, and make the circuit flexible for changing the number of points in the network. The evaluation for finding the shortest path takes one pulse from external clock source. The resultant Figures (4a-4e) show the shortest path for node distribution, Figures (4d-4e) show the reevaluation with new data reach (simulated) for new distribution of path cost values. Figures (5a-5g) represent the hardware implementation of the proposed circuit, where each figures of them in green LEDs shows the points which selected for the shortest path algorithm. There is an important point must be taking in to account, when the number of point in the network map increased up to the limited of the used Xilinx IC then must use another one, which has a large number of gates.

CONCLUSIONS

In this paper the FPGA design of a proposed route optimization circuit has been presented.

This circuit found the shortest route between any two specific points in the network map according to Dijkstra's algorithm. It is helpful because of the flexibility of changing the design by software. Furthermore, it can operate in any frequency according to speed of input data. This route circuit is useful to be used in serves the taxi, security comp. and military applications. Moreover, it can be used to complete the navigation system by connecting a GPS for indicating the mobile user on the map. Also it can be determination a mobile phone number for immediately giving new information to control center.

REFERENCES

[1] M.R.Delavar, F. Samadzadegan and P. Pahlavni, "A GIS-Assisted Optimal Urban Route Finding Approach Based On Genetic Algorithms", Dept. of Surveying and Geomatice, Faculty of Engineering, Unv. Of Tehran, Commission II, Working Group II/5, 2000.

[2] Thamir R.S., Emad S. and Dalia K., "Routing Optimizing of Mobile Network System", Eng. & Tech. Journal, Vol.27, No.8, 2009.

[3] Tatsuya K., Shutaro K., Nobuhiko N. and Ryosuke T, "Optimal Location of Route and Stops of Public Transportation", Proceedings, 6th International Space Syntax Symposium, Istanbul, 2007.

[4] Atulya N. and Hissam T., " A Multi-Criteria Based Approach to Prototyping Urban Road Networks", Issues in Informing Science and Information Technology, Vol. 4, 2007.

[5] Prabhat S. and Sardar P., "Design of Feeder Route Network Using Combined Genetic Algorithm and Specialized Repair Heuristic", Journal of Public Transportation, Vol. 10, No.2, 2007.

[6] Mukti A., B. Srirama, and S.K. Pathan, "Improvement in Transit Service Using GIS-Case Study of Bhavnagar State Transport Depot", Proceesings ESRI National Conference 2005, Held at NOIDA, India.

[7] Shahiar A.Z., and Reza T., "Selecting an Optimum Configuration of Urban One-Way and Two-Way Streets Using Genetic Algorithms", International Journal Of Civil Eng, Vol. 4, No.3, Sept. 2006.

[8] Saffarzadeh, M. and A.M. Broujerdian, "Optimum Route Location Model Considering The Trip Generation Centers and Protected Zones", Iranian Journal of Science & Tech.,Transaction B, Eng. Vol. 31, No.B2, PP. 167-178, 2007.

[9] Tien V. T. Zhao T., " Method and Appartus For Establishing Optimum Route to Intersection Destination for Navigation System", Patent Application Publication, Us 025939 A1, Oct., 2009.

[10] Peter G.H., " Method for Vehicle Route Planning", Patent Application Publication, Us 0010732 A1, Jan., 2010.

Table (1) network node numbers and related path and block values (all values in HEX)

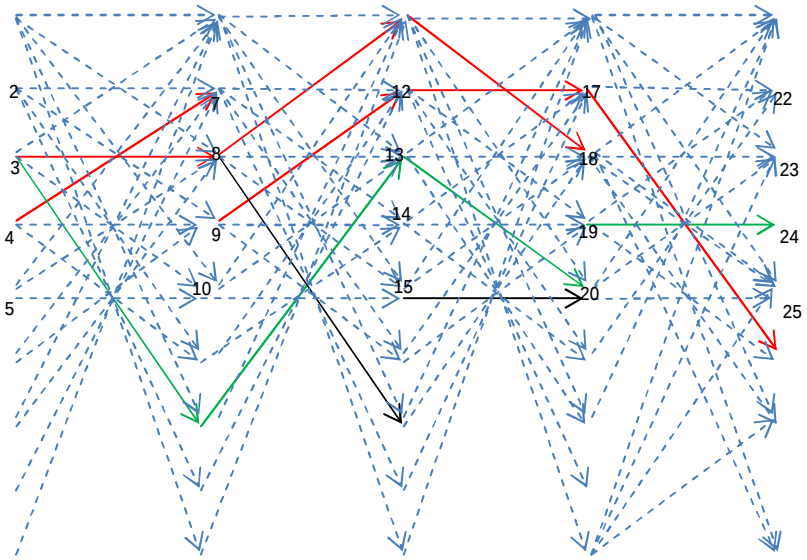
	From	To	Path	Block	PFT	BFT	change
1	1	6	3	0	42603	826	
2		7	4	0	42704	827	
3		8	9	0	42809	828	
4		9	1	0	42901	829	
5		10	11	0	42A0B	82A	
6	2	6	4	0	44604	846	
7		7	12	0	4470C	847	
8		8	6	0	44806	848	
9		9	5	0	44905	849	
10		10	1	0	44A01	84A	
11	3	6	6	0/1	46606	866*	C66
12		7	8/2	1	46708*	C67	46702
13		8	4	1	46804	C68	
14		9	11		4690B	869	
15		10	2/6	1	46A02*	C6A	46A06
16	4	6	8	0	48608	886	
17		7	6	0	48706	887	
18		8	3	0	48803	888	
19		9	5	0	48905	889	
20		10	9	0	48A09	88A	
21	5	6	4	0	4A604	8A6	
22		7	11	0	4A70B	8A7	
23		8	4	0	4A804	8A8	
24		9	13	0	4A90D	8A9	
25		10	5	0	4AA05	8AA	
26	6	11	7	0	4CB07	8CB	
27		12	8	0/1	4CC08	8CC*	CCC
28		13	1	0	4CD01	8CD	
29		14	4	0	4CE04	8CE	
30		15	9	0	4CF09	8CF	
31	7	11	10	1/0	4EB0A	CEB*	8EB
32		12	4	0	4EC04	8EC	
33		13	7	0/1	4ED07	8ED*	CED
34		14	4	0	4EE04	8EE	
35		15	2	0	4EF02	8EF	
36	8	11	8	0	50B08	90B	
37		12	3	1	50C03	D0C	
38		13	9	0	50D09	90D	
39		14	5	0	50E05	90E	
40		15	11	1	50F0B	D0F	
41	9	11	12	0	52B0C	92B	
42		12	1	0	52C01	92C	

COUNT. With Table (1)

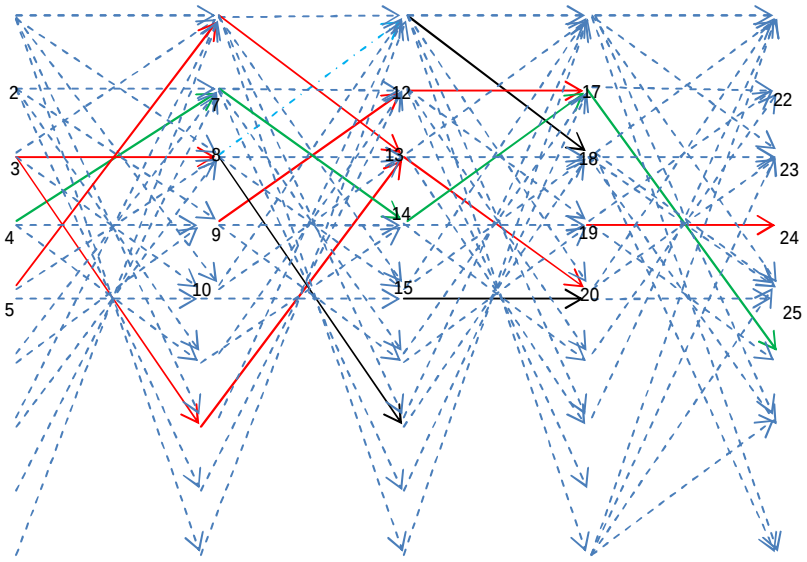
43		13	7	0	52D07	92D	
44		14	5	0	52E05	92E	
45		15	9	0	52F09	92F	
46	10	11	1	0	54B01	94B	
47		12	7	0	54C07	94C	
48		13	4	1	54D04	D4D	
49		14	11	0	54E0B	94E	
50		15	6	0	54F06	94F	
51	11	16	4	0	57004	970	
52		17	8	1	57108	D71	
53		18	11	0	5720B	972	
54		19	12	0	5730C	973	
55		20	6	0	57406	974	
56	12	16	4	0	59004	990	
57		17	9	1	59109	D91	
58		18	12	0	5920C	992	
59		19	8	0	59308	993	
60		20	13	0	5940D	994	
61	13	16	8	0	5B008	9B0	
62		17	14/1	0/1	5B10E*	9B1*	5B101 - DB1
63		18	8	0	5B208	9B2	
64		19	2/6	1	5B302*	DB3	5B306
65		20	3	0	5B403	9B4	
66	14	16	4	0	5D004	9D0	
67		17	5	0	5D105	9D1	
68		18	6	0	5D206	9D2	
69		19	9	0	5D309	9D3	
70		20	2	0	5D402	9D4	
71	15	16	2	0	5F002	9F0	
72		17	2	0	5F102	9F1	
73		18	5	0	5F205	9F2	
74		19	5	0	5F305	9F3	
75		20	7	1	5F407	DF4	
76	16	21	8	0	61508	A15	
77		22	9	0	61609	A16	
78		23	5	0	61705	A17	
79		24	4	0	61804	A18	
80		25	9	0	61909	A19	
81	17	21	3	0	63503	A35	
82		22	10	0	6360A	A36	
83		23	6	0	63706	A37	
84		24	3	1	63803	E38	
85		25	1	0	63901	A39	
86	18	21	8	0	65508	A55	
87		22	6	0	65606	A56	
88		23	3	0	65703	A57	
89		24	8	0	65808	A58	
90		25	1	0	65901	A59	
91	19	21	4	0	67504	A75	
92		22	8	0	67608	A76	
93		23	3	0	67703	A77	
94		24	4	1	67804	E78	

95		25	8	0	67908	A79	
96	20	21	1	0	69501	A95	
97		22	1	0	69601	A96	
98		23	11	0	6970B	A97	
99		24	12	0	6980C	A98	
100		25	6	0	69906	A99	

Note: the sign (*) with the values mean its values will be changed by newer one as in column under title [change].



a- Network node paths



b- Network node paths with new values

Figure (1) Network node paths map

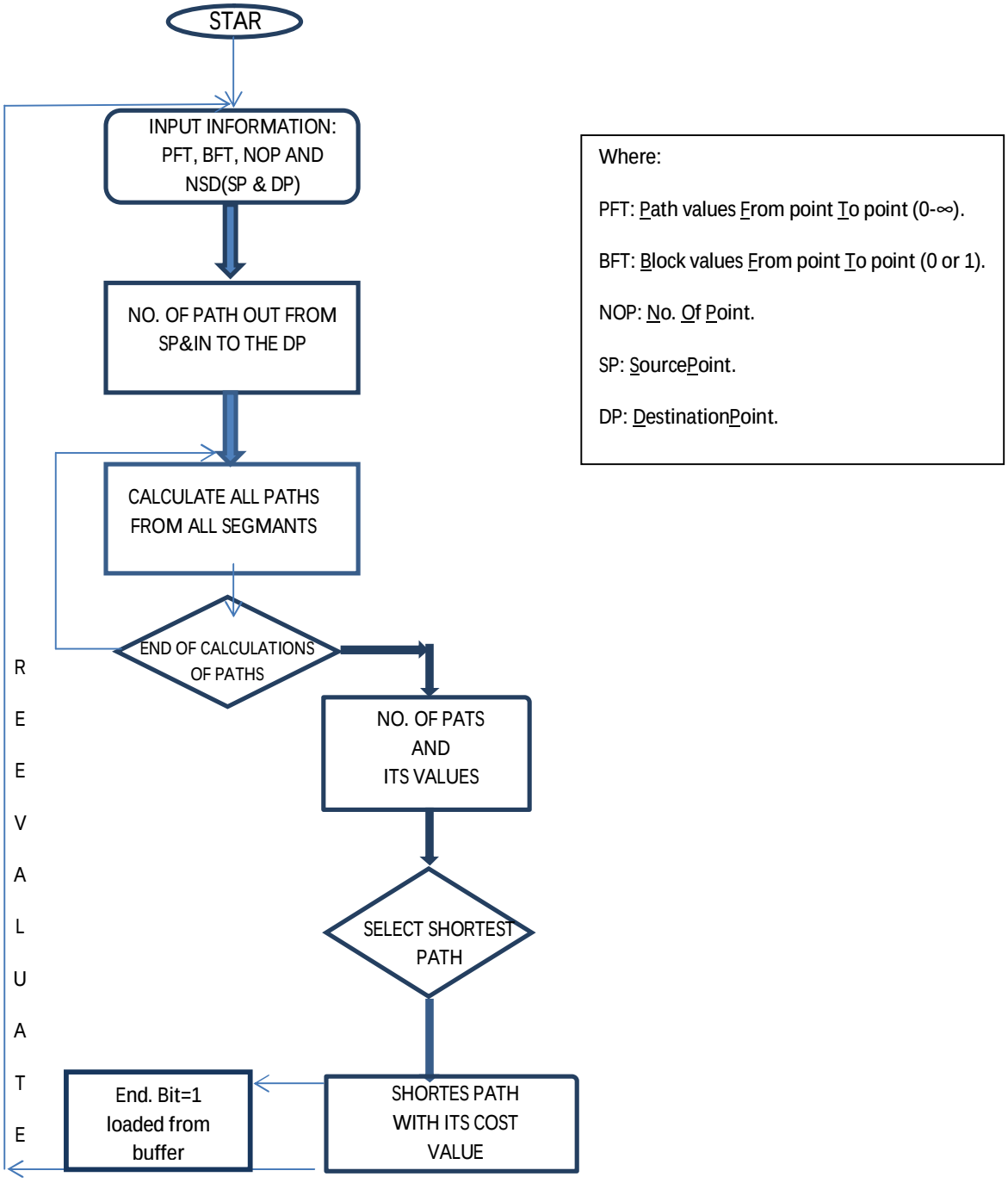


Figure (2) flowchart of implemented algorithm

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity node is
        port(clock:instd_logic;
                pft:instd_logic_vector(18 downto 0);
                bft:instd_logic_vector(11 downto 0);
                nsd:instd_logic_vector(10 downto 0);
                n1:outstd_logic_vector(4 downto 0);
                n2:outstd_logic_vector(4 downto 0);
                n3:outstd_logic_vector(4 downto 0);
                n4:outstd_logic_vector(4 downto 0);
                n5:outstd_logic_vector(4 downto 0);
                plength:outstd_logic_vector(11 downto 0)
                );
end node;
architecture Behavioral of node is
        type xr1 is array (1 to 20) of integer;
        typexr is array (1 to 5) of xr1;
        type xro1 is array (1 to 5) of integer;
        typexro is array (1 to 5) of xro1;
        typexpath is array (1 to 125) of xro1;
        type xp1 is array (1 to 2) of integer;
        typexp is array (1 to 125) of xp1;
begin

        ifclock'event and clock='1' then
                if check=0 then
                        for i in 1 to n loop
                        for j in 1 to n*(n-1) loop
                                rm(i)(j):=0;
                                bm(i)(j):=0;
                        end loop;
                        end loop;
                end if;
        end if;

        ifclock'event and clock='1' and pft(18)='1' then
                rmr:=CONV_INTEGER(pft(12 downto 8));
                rmc:=CONV_INTEGER(pft(17 downto 13));
                whilermr> 5 loop
                rmr:=rmr-5;
                end loop;
                rm(rmr)(rmc):=CONV_INTEGER(pft(7 downto 0));
                check:=check+1;
        end if;
        ifnonbsd=0 then
        ind:=0;
                ind:=ind+1;
                path(ind)(1):=r(snrow)(sncol);
                path(ind)(2):=r(dnrow)(dncol);
        end if;
        ifnonbsd=1 then
                ind:=0;
                for i in 1 to n loop
                        ind:=ind+1;
                        path(ind)(1):=r(snrow)(sncol);
                        path(ind)(2):=r(i)(sncol+1);
                        path(ind)(3):=r(dnrow)(dncol);
                end loop;
        end if;
        ifnonbsd=2 then
                ind:=0;
                for i in 1 to n loop
                        for j in 1 to n loop
                                ind:=ind+1;
                                path(ind)(1):=r(snrow)(sncol);
                                path(ind)(2):=r(i)(sncol+1);
                                path(ind)(3):=r(j)(sncol+2);
                                path(ind)(4):=r(dnrow)(dncol);
                        end loop;
```

Identify I/O data

Identify matrixes

Multiply the matrixes

Enter the path according to the block

Calculate the node from source to the destination points

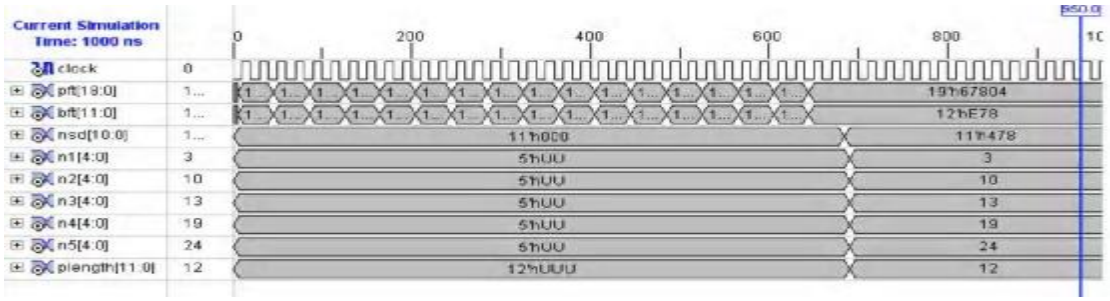
```
                    end loop;
                end if;
                ifnonbsd=3 then
                    ind:=0;
                    for i in 1 to n loop
                        for j in 1 to n loop
                            for k in 1 to n loop
                                ind:=ind+1;
                                path(ind)(1):=r(snrow)(sncol);
                                path(ind)(2):=r(i)(sncol+1);
                                path(ind)(3):=r(j)(sncol+2);
                                path(ind)(4):=r(k)(sncol+3);
                                path(ind)(5):=r(dnrow)(dncol);
                            end loop;
                        end loop;
                    end loop;
                end if;
                ind:=0;

                for i in 1 to n loop
                    final(i):=0;
                end loop;

                ifind=0 then
                    sm:=0;
                else
                    pathind:=p(1)(1);sm:=p(1)(2);
                        for i in 1 to ind loop
                            ifsm> p(i)(2) then
                                sm:=p(i)(2);
                                pathind:=p(i)(1);
                            end if;
                        end loop;
                        for i in 1 to noppn+1 loop
                            final(i):=path(pathind)(i);
                        end loop;
                end if;
                end if;
        end process;
end Behavioral;
```

Count. With above

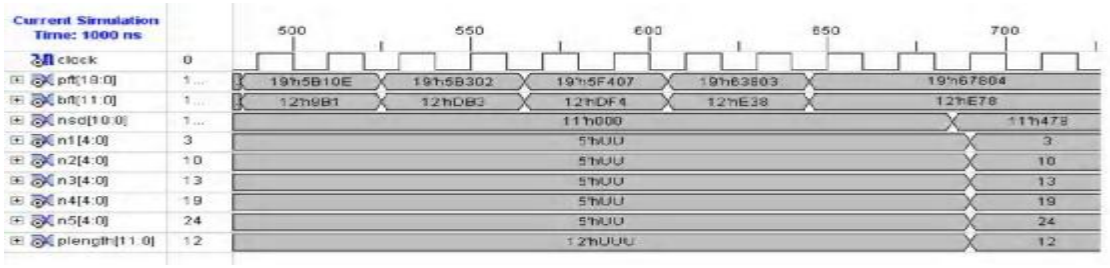
Calculate a shortest path

NOTE: This is the important statement of program (for reduce the no. of pages).

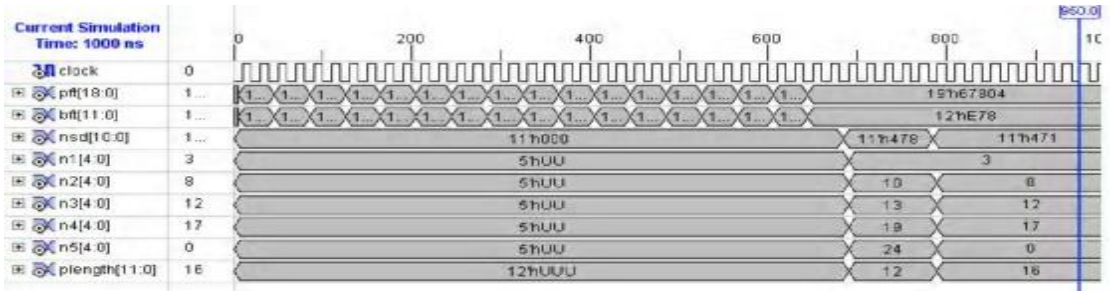
Figure (3) VHDL program implementation of Xilinx (spartn-3 XC3S200)



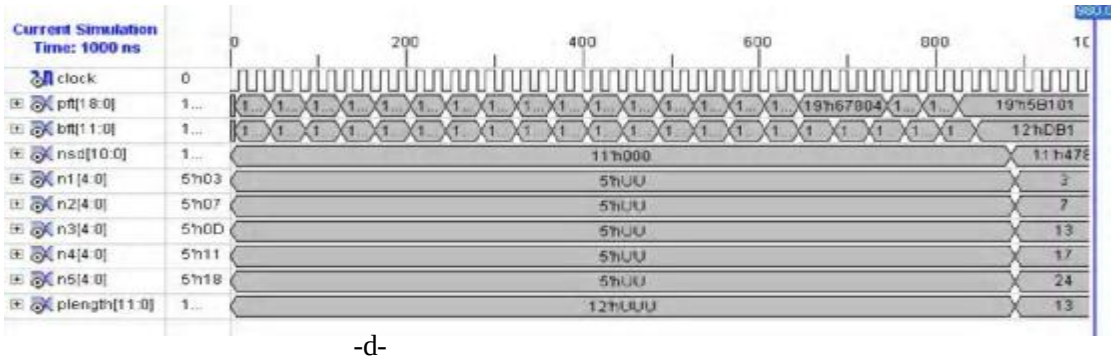
-a-



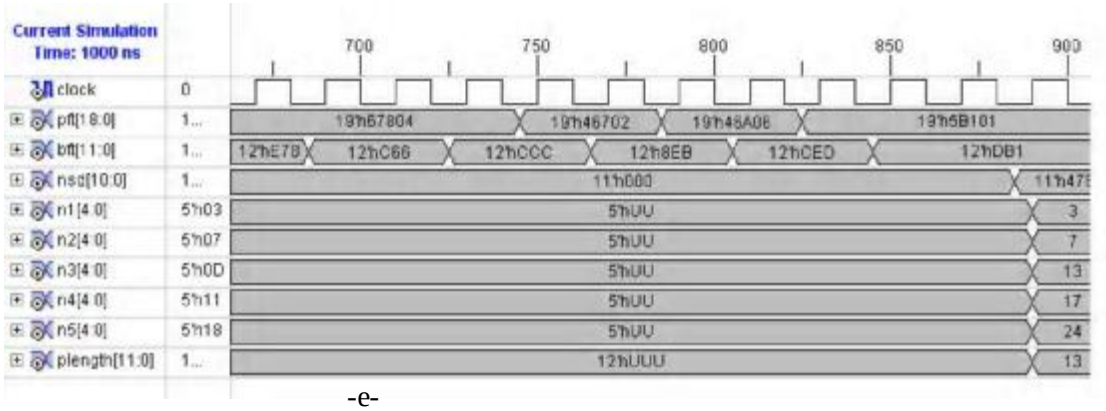
-b-



-c-



-d-



-e-

Figure (4) Result of the implementation

a- First point (3) b- Second point (10) c- Third point (13)

e- Fourth point (19) f- Fifth point (24) g- Path cost (12)

Figure (5) hardware implementation of proposed circuit