UDC 004.3

LUT BASED FREDKIN GATE

Tyurin S. F. – Dr. Sc., Professor, Professor of the Automation and Telemechanic Department, Perm National Research Polytechnic University, Perm, Russia; Professor of the Software Computing Systems Department, Perm State University, Perm, Russia.

ABSTRACT

Context. The concept of existing computers when achieving nanoscale hardware has almost exhausted itself. This also applies to computing power and related energy costs. Reversible computing, for example billiard-ball computer, is the base model of the quantum computing which are considered to be the prospect of IT technology. Billiard-ball computing is energy-effective computing or green computing. Base of such paradigm are special logic gates. However, the mathematical apparatus for creating such computers has not yet been fully developed. The problem is that for new reversible elements that have a one-to-one correspondence between inputs and outputs, the application of well-known methods of analysis and synthesis encounters certain difficulties. So, for example, it is forbidden to use branching, which significantly complicates the synthesis. Reversible elements should provide signal transmission in the forward and reverse directions, which is in principle feasible in binary logic based on tri-states buffers, but significantly complicates the device, increases the crystal area and power consumption, which they are designed to reduce.

Objective. The goal of the work is the analysis of the functionally complete reversible gates that named Toffoli gate, Fredkin gate, the analysis of the binary full adder, based on Fredkin gates and to design method for circuits based proposed gate.

Methods. Analysis of the digital circuits with Boolean algebra. Synthesis digital circuits with proposed decomposition method. Design Fredkin gate in term of the FPGA's Look up Table. Simulation of proposed element in the system NI Multisim by National Instruments Electronics Workbench Group.

Results. Analysis of the full adder based on Fredkin gates. Synthesis method of the reversible circuits based on Fredkin gates. LUT based Fredkin gate and it simulation.

Conclusions. The conducted studies allows us to build circuits based on Fredkin gates from proposed novel elements.

KEYWORDS: Quantum Computing, Logic Function, Fredkin Gate, Shannon decomposition or Boolean factorization.

ABBREVIATIONS

CNF is a Conjunctive Normal Form;
DNF is a Disjunctive Normal Form;
FG is a Fredkin Gate;
FPGA is a Field-Programmable Gate Array;
LUT is a Look up Table of FPGAs;
TG is a Toffoli Gate;
XOR is an exclusive OR.

NOMENCLATURE

A is a third Fredkin Gate (or Toffoli Gate) input;
B is a second Fredkin Gate (or Toffoli Gate) input;
C is a first Fredkin Gate (or Toffoli Gate) input;
$F1$ is a first Fredkin Gate output;
$F2$ is a second Fredkin Gate output;
$F3$ is a third Fredkin Gate input;
g is a "trash" output of an Adder;
$j(x_1x_2,...x_n), j = f,h,g,v,w...$ is a decomposition function;
k is an iteration variable;
p is a first input/output of an Adder;
q is a second input/output of an Adder;
r is a third input or input carry of an Adder;
S is an additional function;
$x_1x_2,...x_n$ are the abstract variables;
$Z1$ is a first Toffoli Gate output;
$Z2$ is a second Toffoli Gate output;
$Z3$ is a third Toffoli Gate output.

INTRODUCTION

Quantum computing is actual area of modern science and technology [1, 2]. It is believed that quantum computers, that manipulates q-bits and q-bytes, can give a sharp leap in the memory size and computing power to solve many IT problems in the future. The area of reversible computation is closely related to these studies [3]. Reversible computation means that we can reverse calculation process and get input data, for example to check out computation. Reversible computation is the base of green computing. For example, so called "billiard-ball" computing get energy quants ("balls"), in contrast traditional computing, only online, so power consumption is many less. In ideal model the same balls can used many times and turn back to the source. In Quantum computer, in special quantum logic balls are q-bits, but this paradigm can be successfully used in binary logic, binary computers and digital circuit. As energy quanta in billiard computers, you can use charges stored on capacitors. In addition, it can be used for modeling in quantum devices design. Reversible computing requires novel reversible gates, for example binary gates. Some of these elements are TG, FG. Analyze and synthesis reversible gates circuits have their own characteristics and difficulties, for example fan-out problem. However, fan-out restricted allows solving race hazard problem in digital devices. Therefore, a detailed examination of these features is of considerable interest.

The object of study are the elementary Quantum gates and circuits based on binary Fredkin gates [4, 5].

The purposes of the work are to analyze of the full adder based on binary Fredkin gates and design binary Fredkin gates circuits synthesis method based on proposed element, similar LUT FPGA.

1 PROBLEM STATEMENT

Given: Full adder based on FG [9]. There are five rows of signals p,q,r, including constants 0,1 and five reversible elements. The internal structure of the Fredkin

© Tyurin S. F., 2020
DOI 10.15588/1607-3274-2020-1-5

element is not disclosed. The first p and second q are the bits ("q-bits" in common case) to be added, the third r is the input carry (carry in). Some of the outputs (p,q) of the device repeat the inputs (p,q). There are outputs parity $r \oplus p \oplus q$ and carry out $rp \vee rq \vee pq$. There is "trash" output g. Every Fredkin gate has own inputs C,B,A and outputs $F1,F2,F3$.

In case C=1, $F1$=1, $F2=A$, $F3=B$. In case C=0, $F1$=0,$F2=B$,$F3=A$. In the literature [1–17], the problems of analyze and synthesis FG-circuits, FG realization in FPGA are not fully covered. There is no detailed description in terms of Boolean algebra [18].

It is required: perform Boolean analyze of the FG-full adder, to propose Boolean synthesis method FG-circuit for a given logic function, perform to design LUT based FG and it simulation.

Describe every Fredkin gate input like Boolean function $A_i(p,q,r), B_i(p,q,r), C_i(p,q,r), i = 1...5..$

Describe every Fredkin gate output like Boolean function $F_{1.i}(p,q,r), F_{2.i}(p,q,r), F_{3.i}(p,q,r)$.

Prove that the required parity and carry are formed at the device outputs.

Based on the analysis, to propose a synthesis method for a given logical function and verify it by completing the construction of the circuit from the end. Taking logic function $f(x_1x_2,...x_n)$ to get

$$A_i(x_1x_2,...x_n), B_i(x_1x_2,...x_n), C_i(x_1x_2,...x_n), i;$$
$$F_{1.i}(x_1x_2,...x_n), F_{2.i}(x_1x_2,...x_n), F_{3.i}(x_1x_2,...x_n).$$

It is need to design element's architecture based on LUT FPGA logic elements and to perform simulate the proposed element.

2 REVIEW OF THE LITERATURE

Currently rises research in reversible computing, in quantum computing [1–5]. For the quantum operations special gates are used. Let us consider how they are described in the literature [1–9] and obtain in detail the corresponding Boolean functions.

Toffoli gate TG [9], proposed in 1980, is Control-Control NOT or CCNOT. Toffoli gate's conditional symbol and functions shows Fig. 1 and Table 1.

Minimization of the Toffoli gate's function $Z3$ by Karnaugh map and permutation matrix shows Fig. 1.

Here C,B are control inputs (Fig.1). If $C=B$=1 gate works like NOT gate. Otherwise, the repeat function of A input is implemented on the $Z3$ output. Expressions (1) describes Toffoli gate in DNF:

$$\begin{cases} Z_1(CBA) = C; \\ Z_2(CBA) = B; \\ Z_3(CBA) = \overline{C}A \vee \overline{B}A \vee CB\overline{A}. \end{cases} \quad (1)$$

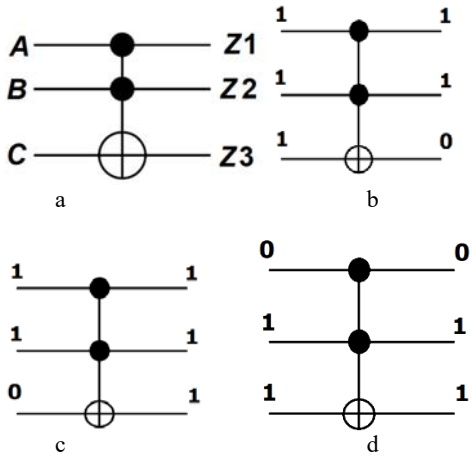
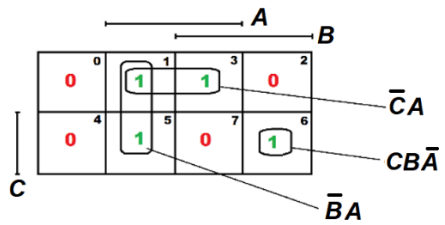


Figure 1 – Toffoli gate: a – Conditional Symbol, A,B,C – Inputs, $Z1,Z2,Z3$ – outputs; b – $C=B$=1,A=1,$Z3$=0; c – $C=B$=1,A=0,$Z3$=1; d – if C not equal B, then $Z3$=A

Table 1 – Toffoli Gate's Truth Table for the binary logic

№	C	B	A	Z1	Z2	Z3
0	0	0	0	0	0	0
1	0	0	1	0	0	1
2	0	1	0	0	1	0
3	0	1	1	0	1	1
4	1	0	0	1	0	0
5	1	0	1	1	0	1
6	1	1	0	1	1	1
7	1	1	1	1	1	0



a

1	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0
0	0	1	0	0	0	0	0
0	0	0	1	0	0	0	0
0	0	0	0	1	0	0	0
0	0	0	0	0	1	0	0
0	0	0	0	0	0	0	1
0	0	0	0	0	0	1	0

b

Figure 2 – Toffoli gate: a – Truth table as the Karnaugh map; b – Permutation Matrix

Note that $Z_3(C1A) = \overline{C}A \vee C\overline{A} = C \oplus A$.

Fredkin gate FG (CSWAP, Controlled swap gate) proposed supposedly in 1982 together with Toffoli [1–5, 9]. Fredkin gate's Conditional Symbol and functions shows Fig. 3 and Table 2.

Minimization of the Fredkin gate's functions $Z2,Z3$ by Karnaugh maps and permutation matrix shows Fig.4

© Tyurin S. F., 2020
DOI 10.15588/1607-3274-2020-1-5

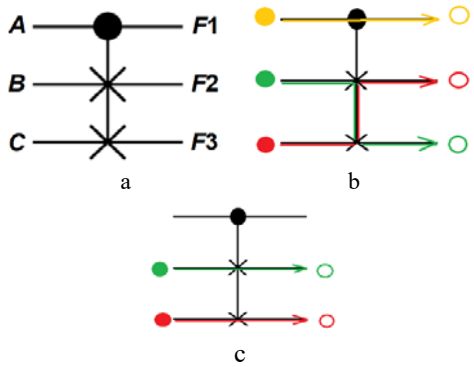
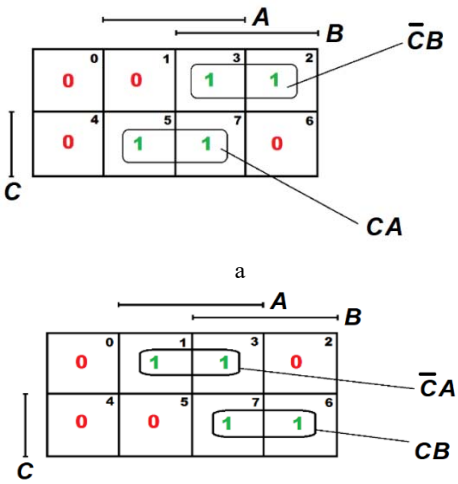


Figure 3 – Fredkin Gate: a – Conditional Symbol; b – input green and red balls swapped if C=1 (yellow ball); c – input green and red balls transit to the same outputs if C=0

Table 2 – Fredkin Gate's Truth Table for the binary logic

№	C	B	A	F1	F2	F3
0	0	0	0	0	0	0
1	0	0	1	0	0	1
2	0	1	0	0	1	0
3	0	1	1	0	1	1
4	1	0	0	1	0	0
5	1	0	1	1	1	0
6	1	1	0	1	0	1
7	1	1	1	1	1	1



1	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0
0	0	1	0	0	0	0	0
0	0	0	1	0	0	0	0
0	0	0	0	1	0	0	0
0	0	0	0	0	0	1	0
0	0	0	0	0	1	0	0
0	0	0	0	0	0	0	1

c

Figure 4 – Fredkin gate: a – Truth table as the Karnaugh map; b – Permutation Matrix

Note that the parity is observed in Truth table Fig.4 a: the number of units is the same at the input and output, unlike Table 1. Really, in billiard logic it is impossible to change the number of balls.

Therefore in DNF and Jegalkin Form (Polinom) present expressions (2):

$$\begin{cases} F_1 = C; \\ F_2 = CA \vee \overline{C}B = CA \oplus \overline{C}B = \\ \overline{(CA)}\overline{C}B \vee (CA)(\overline{\overline{C}B}) = (\overline{C} \vee \overline{A})\overline{C}B \vee (CA)(C \vee \overline{B}) = \\ = CA \vee \overline{C}B; \\ F_3 = \overline{C}A \vee CB = \overline{C}A \oplus CB = \\ = \overline{(\overline{C}A)}CB \vee (\overline{C}A)(\overline{\overline{C}B}) = (C \vee \overline{A})CB \vee (\overline{C}A)(\overline{C} \vee \overline{B}) = \\ = CA \vee \overline{C}B. \end{cases} \quad (2)$$

In $F3$ C and not C swapped. Let S is additional Function [9]:

$$S(CBA) = (A \oplus B)C. \quad (3)$$

Therefore, we can get expression (4).

$$\begin{aligned} F_2(CBA) &= B \oplus S, \\ F_1(CBA) &= A \oplus S. \end{aligned} \quad (4)$$

Poretsky's law used $C\overline{B} \vee B = C \vee B$ and common gluing law $AC \vee AB \vee \overline{C}B = CA \vee \overline{C}B$ – expression (5):

$$\begin{aligned} &(A \oplus B)C \oplus B = \\ &= (CA\overline{B} \vee C\overline{A}B)\overline{B} \vee \overline{(CA\overline{B} \vee C\overline{A}B)}B = \\ &= CA\overline{B} \vee (\overline{C} \vee \overline{A} \vee B)(\overline{C} \vee A \vee \overline{B})B = \\ &= CA\overline{B} \vee (\overline{C} \vee \overline{A}\overline{B} \vee AB)B = \\ &= CA\overline{B} \vee \overline{C}B \vee AB = \\ &= A(C\overline{B} \vee B) \vee \overline{C}B = \\ &= A(C \vee B) \vee \overline{C}B = \\ &= AC \vee AB \vee \overline{C}B = \\ &= CA \vee \overline{C}B. \end{aligned} \quad (5)$$

Similarly, we can prove next:

$$\begin{aligned} &(A \oplus B)C \oplus A = \\ &= (CA\overline{B} \vee C\overline{A}B)\overline{A} \vee \overline{(CA\overline{B} \vee C\overline{A}B)}A = \\ &= C\overline{A}B \vee (\overline{C} \vee \overline{A} \vee B)(\overline{C} \vee A \vee \overline{B})A = \\ &= C\overline{A}B \vee (\overline{C} \vee \overline{A}\overline{B} \vee AB)A = \\ &= C\overline{A}B \vee \overline{C}A \vee AB = \\ &= B(C\overline{A} \vee A) \vee \overline{C}A = \\ &= B(C \vee A) \vee \overline{C}A = \\ &= \overline{C}A \vee CB \vee AB = \\ &= CB \vee \overline{C}A. \end{aligned} \quad (6)$$

Functional completeness not respected for $F2$,$F3$. $F2$ has number 172 and not corresponds Post criterions (exist truth-preserving and falsity-preserving: saves constants 0.1). F3 has number 202 and not corresponds Post criterions too (saves constants 0.1). When B=0 for $F2$, we get binary number 1000 (conjunction). Conjunction implementation shows expression (7):

© Tyurin S. F., 2020
DOI 10.15588/1607-3274-2020-1-5

$$F_2(CBA) = CA \vee \overline{C}B \Rightarrow F_2(C0A) = CA. \quad (7)$$

When B=1 for $F3$, we get binary number 1110 (disjunction). Implementation of Disjunction shows expression (8):

$$F_3(CBA) = CB \vee \overline{C}A \Rightarrow F_3(C1A) = CB \vee \overline{C}A = \\ = C \vee \overline{C}A = C \vee A. \quad (8)$$

When B=1,A=0 for $F2$, we get binary number 10 (NOT). Implementation of NOT operation shows expression (9):

$$F_2(CBA) = CA \vee \overline{C}B \Rightarrow F_2(C10) = \overline{C}. \quad (9)$$

Thus, a Boolean analysis of TG and FG expressions given in the literature performed. Expressions (7), (8), (9) describes minimal functional complete sets AND,NOT (7), (9) or OR,NOT (8),(9).

3 MATERIALS AND METHODS

Let us use Boolean analyze method for the circuit of a single-bit binary adder based on five Fredkin gates [9] – Fig. 5.

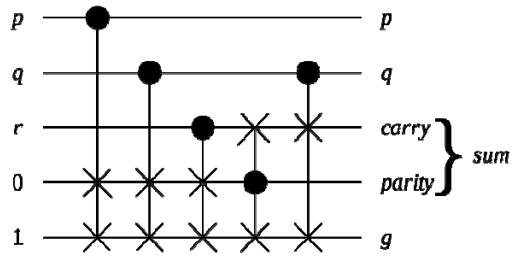


Figure 5 – Full adder based on five FG

Note that there are no branches in this circuit. They are forbidden in reversible logic. Instead, branches are used repeaters. On the Fig. 5 uses additionally constant "0" and "1" (ancilla bits).

We get the functions at the output of the first element (the inputs "p", 0,1 to the left of the vertical bar, the outputs F). We get the expression (10):

$$\begin{cases} F_1(CBA) = C \Rightarrow F_{1.1}(p,0,1) = p; \\ F_2(CBA) = CA \vee \overline{C}B \Rightarrow F_{1.2}(p,0,1) = p1 \vee \overline{p}0 = p; \\ F_3(CBA) = \overline{C}A \vee CB \Rightarrow F_{1.3}(p,0,1) = \overline{p}1 \vee p0 = \overline{p}. \end{cases} \quad (10)$$

So we have outputs of first gate – Fig. 6.

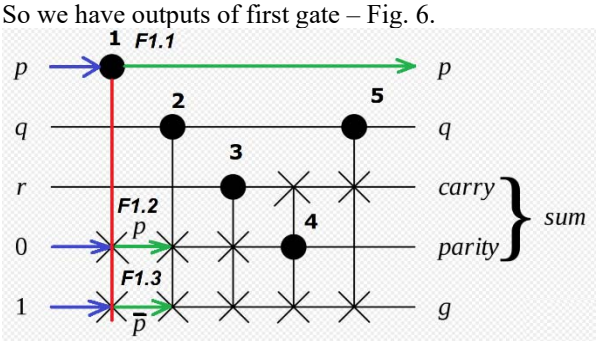


Figure 6 – Analysis of first FG

Then it is necessary to get outputs of second gate — expression (11):

$$\begin{cases} F_1(CBA) = C \Rightarrow F_{2.1}(q,p,\overline{p}) = p; \\ F_2(CBA) = CA \vee \overline{C}B \Rightarrow F_{2.2}(q,p,\overline{p}) = q\overline{p} \vee \overline{q}p = q \oplus p; \\ F_3(CBA) = \overline{C}A \vee CB \Rightarrow F_{2.3}(q,p,\overline{p}) = \overline{q}\,\overline{p} \vee qp = \overline{q \oplus p}. \end{cases} \quad (11)$$

In terms of the Fredkin gate, it sounds like this: bits 0 and 1 are swapped places if the control signal = 1 (is set), that is, when p = 1, signal 0 appears at the output of $F1.2$, and at the output of $F1.3$ appears 1. These crosses mean "swap". Analyze of the second element shows Fig. 7.

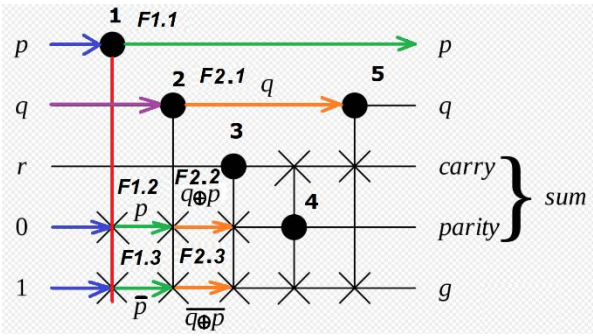


Figure 7 – Analysis of second FG

Thus, at the outputs of $F2.2$, and at the output of $F2.3$ the input bits 0 and 1 again change places if p = q. We analyze the third element:

$$\begin{cases} F_1(CBA) = C \Rightarrow F_{3.1}(r,(q \oplus p),(\overline{q \oplus p})) = r; \\ F_2(CBA) = CA \vee \overline{C}B \Rightarrow F_{3.2}(r,(q \oplus p),(\overline{q \oplus p})) = r(\overline{q \oplus p}) \vee \overline{r}(q \oplus p) = \\ \qquad = r \oplus p \oplus q; \\ F_3(CBA) = \overline{C}A \vee CB \Rightarrow F_{3.3}(r,(q \oplus p),(\overline{q \oplus p})) = \overline{r}(\overline{q \oplus p}) \vee r(q \oplus p) = \\ \qquad = \overline{r \oplus p \oplus q}. \end{cases} \quad (12)$$

Thus, we get Fig. 8.

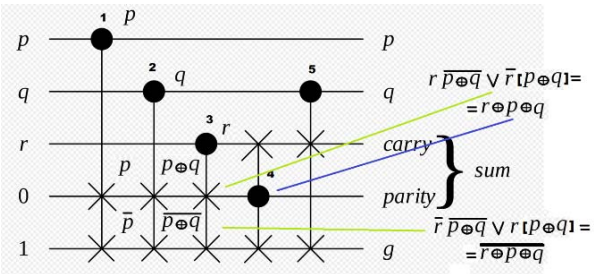


Figure 8 – Analysis of third FG

We see that indeed at the second output of the third element a sum or a sign of parity is formed, in fact – not parity. The control signal is the sum $r \oplus p \oplus q$ because it passes in transit to the output (parity). Other signals are r and $\overline{r \oplus p \oplus q}$. So forth element implements the following functions:

© Tyurin S. F., 2020
DOI 10.15588/1607-3274-2020-1-5

$$\begin{cases} F_1(CBA) = C \Rightarrow F_{4.1}(\{r \oplus p \oplus q\}, r, \{\overline{r \oplus p \oplus q}\}) = \\ \quad = r \oplus p \oplus q; \\ F_2(CBA) = CA \vee \overline{C}B \Rightarrow \\ F_{4.2}(\{r \oplus p \oplus q\}, r, \{\overline{r \oplus p \oplus q}\}) = \\ (r \oplus p \oplus q)(\overline{r \oplus p \oplus q}) \vee (\overline{r \oplus p \oplus q})r = \\ \quad = (\overline{r \oplus p \oplus q})r; \\ F_3(CBA) = \overline{C}A \vee CB \Rightarrow \\ F_{4.3}(\{r \oplus p \oplus q\}, r, \{\overline{r \oplus p \oplus q}\}) = \\ (\overline{r \oplus p \oplus q})(\overline{r \oplus p \oplus q}) \vee (r \oplus p \oplus q)r = \\ \quad = (\overline{r \oplus p \oplus q}) \vee (r \oplus p \oplus q)r = \\ \quad = (\overline{r \oplus p \oplus q}) \vee r. \end{cases} \tag{13}$$

The fifth element implements the following functions

$$\begin{cases} F_1(CBA) = C \Rightarrow F_{5.1}(q, \{(\overline{r \oplus p \oplus q})r\}, \{(\overline{r \oplus p \oplus q}) \vee r\}) = \\ \quad = q; \\ F_2(CBA) = CA \vee \overline{C}B \Rightarrow F_{5.2}(q, \{(\overline{r \oplus p \oplus q})r\}, \{(\overline{r \oplus p \oplus q}) \vee r\}) = \\ q\{(\overline{r \oplus p \oplus q}) \vee r\} \vee \overline{q}(\overline{r \oplus p \oplus q})r = \\ \quad = q(\overline{r \oplus p \oplus q}) \vee qr \vee r(\overline{r \oplus p \oplus q}); \\ F_3(CBA) = \overline{C}A \vee CB \Rightarrow \\ F_{5.3}(q, \{(\overline{r \oplus p \oplus q})r\}, \{(\overline{r \oplus p \oplus q}) \vee r\}) = \\ \overline{q}\{(\overline{r \oplus p \oplus q}) \vee r\} \vee q\{(\overline{r \oplus p \oplus q})r\} = \\ \quad = \overline{pq} \vee r\overline{q} \vee r\overline{p} = \overline{r(pq)} \vee r(\overline{p \vee q}). \end{cases} \tag{14}$$

Therefore $q(\overline{r \oplus p \oplus q}) \vee qr \vee r(\overline{r \oplus p \oplus q})$ – is carry, $\overline{r(pq)} \vee r(\overline{p \vee q})$ – is garbage function. It was analysis.

Now let us solve the inverse problem. Author proposes synthesis method for FGs circuits with single output $f(x_1x_2,...x_n)$, based on decomposition by *A,B,C* forms (2). First step is selection some variable x_i:

$$\begin{aligned} F_{k.2} &= f(x_1x_2,...x_n) = \\ &= x_i g(x_1x_2,...x_n) \vee \overline{x}_i h(x_1x_2,...x_n). \end{aligned} \tag{15}$$

Therefore, from (15) we can get inputs of last FG:

$$\begin{aligned} C_k &= x_i; \\ B_k &= h(x_1x_2,...x_n); \\ A_k &= g(x_1x_2,...x_n). \end{aligned} \tag{16}$$

Now regard some input (*A* or *B*), for example, *B* of last FG (16) and select new variable x_j:

$$\begin{aligned} B_k &= h(x_1x_2,...x_n) = x_j v(x_1x_2,...x_n) \vee \\ &\vee \overline{x}_j w(x_1x_2,...x_n). \end{aligned} \tag{17}$$

So, we have inputs of *k*–1 FG. If $x_j \neq x_i$ it means that x_i – is output of another FG, not *k*–1 FG, x_j – is output of all circuit. Similar steps are performed until the decomposition functions (*g,h,v,w*... etc.) become input variables $x_1x_2,...x_n$ or constants 0.1.

4 EXPERIMENTS

Now regard synthesis. Let we need to get circuit for next function on *F*2 output of the last FG:

$$\overline{q}(\overline{r \oplus p \oplus q}) \vee \overline{q}r \vee \{(\overline{r \oplus p \oplus q})r. \tag{18}$$

It is carry function (12). Conditional FG symbol shows Fig. 9.

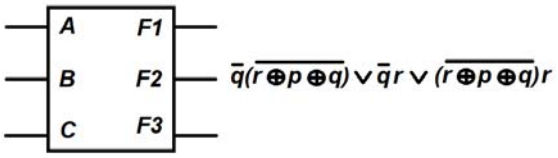


Figure 9 – Last FG

Let use Shannon decomposition or Boolean factorization to get *A,B,C* functions. Select *q* variable:

$$\begin{aligned} &\overline{q}\left(\overline{q}(\overline{r \oplus p \oplus q}) \vee \overline{q}r \vee \{(\overline{r \oplus p \oplus q})r\right) \vee \\ &\vee q\left(\overline{q}(\overline{r \oplus p \oplus q}) \vee \overline{q}r \vee \{(\overline{r \oplus p \oplus q})r\right). \end{aligned} \tag{19}$$

Conversing (19) we can get expression (20):

$$\begin{aligned} &\overline{q}\left((\overline{r \oplus p \oplus q}) \vee r \vee \{(\overline{r \oplus p \oplus q})r\right) \vee q\left(\{(\overline{r \oplus p \oplus q})r\right) = \\ &= \overline{q}\left((\overline{r \oplus p \oplus q}) \vee r\right) \vee q\left(\{(\overline{r \oplus p \oplus q})r\right). \end{aligned} \tag{20}$$

Expression (20) means, that inputs FG(*k*) are next:

$$\begin{aligned} C &= q; \\ B &= (\overline{r \oplus p \oplus q})r; \\ A &= (\overline{r \oplus p \oplus q}) \vee r. \end{aligned} \tag{21}$$

Expression (21) describes outputs of FG(*k*–1) too. So we have FG(*k*) circuit, shows at Fig. 10.

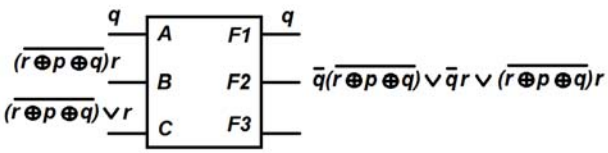


Figure 10 – FG(*k*) circuit

Having inputs *A,B,C* we can get *F*3, but we don't want. Output *F*3 is not needed (trash). Getting inputs of FG(*k*–1) shows factorization by *r*:

$$\begin{aligned} F_{k-1.2} &= r\left((\overline{r \oplus p \oplus q})r\right) \vee \overline{r}\left((\overline{r \oplus p \oplus q})r\right) = \\ &= (\overline{r \oplus p \oplus q})r. \end{aligned} \tag{22}$$

Expression (22) means, that *q* to the FG(*k*) transit from another gate:

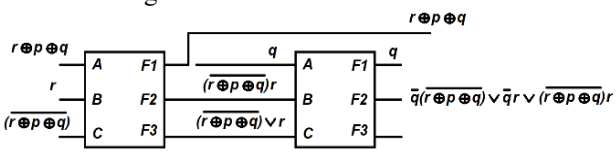


Figure 11 – FG(*k*) + FG(*k*–1) circuit

Further, we consider $r \oplus p \oplus q$ (*C* input) and *r* – factorization:

© Tyurin S. F., 2020
DOI 10.15588/1607-3274-2020-1-5

$$
\begin{aligned}
& r(r \oplus p \oplus q) \vee \bar{r}(r \oplus p \oplus q) = \\
& = r\left(r\bar{q}\bar{p} \vee rqp \vee r\bar{q}p \vee \bar{r}q\bar{p}\right) \vee \\
& \vee \bar{r}\left(r\bar{q}\bar{p} \vee rqp \vee r\bar{q}p \vee \bar{r}q\bar{p}\right) = \\
& = r(\bar{q}\bar{p} \vee qp) \vee \bar{r}(\bar{q}p \vee q\bar{p}) = \\
& = r(\overline{p \oplus q}) \vee \bar{r}(p \oplus q).
\end{aligned} \tag{23}
$$

Expression (23) means that k–2 FG inputs are formulas (24):

$$
\begin{aligned}
C &= r; \\
B &= p \oplus q; \\
A &= \overline{p \oplus q}.
\end{aligned} \tag{24}
$$

So Fig. 12 shows a new circuit.

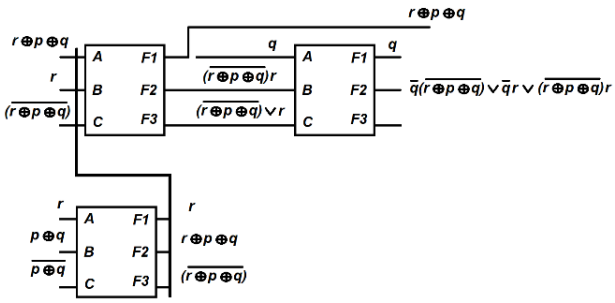


Figure 12 – FG(k) + FG(k–1) + FG(k–2) circuit

Next step is p-factorization:

$$
\begin{aligned}
& p(p \oplus q) \vee \bar{p}(p \oplus q) = \\
& = p\left(\bar{p}q \vee p\bar{q}\right) \vee \bar{p}\left(\bar{p}q \vee p\bar{q}\right) = \\
& = p\bar{q} \vee \bar{p}q.
\end{aligned} \tag{25}
$$

Fig. 13 shows next circuit according additional expression (25).

At last, we consider q – factorization and get $F_{k-4}(q,1,0)$. So full circuit is shown at Fig. 14.

We see that the inputs of the device are r, p, q, 0, 1 and, although we got a slightly different Fig. 5 circuit, it implements the same functions, namely a full single-bit adder. The output of the last gate $F3$ is not used. Output $r \oplus p \oplus q$ is parity (sum). It is easy to see that the circuit is reversible.

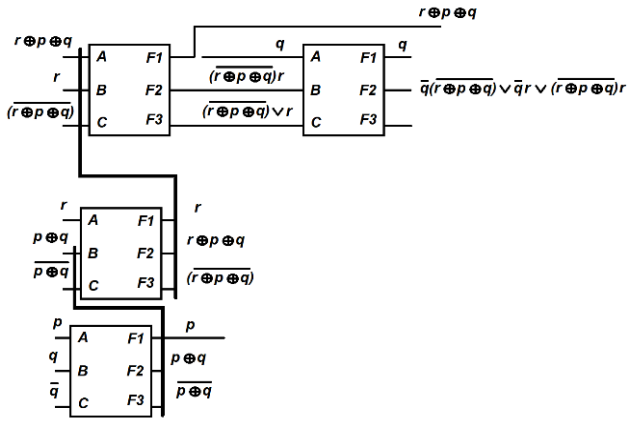


Figure 13 – FG(k) + FG(k–1)+ FG(k–2) + FG(k–3) circuit

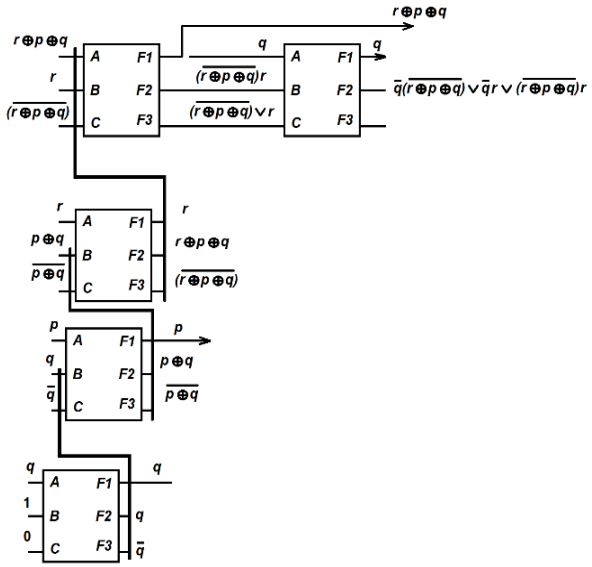


Figure 14 – Full FG(k) + FG(k–1)+ FG(k–2) + FG(k–3) + +FG(k–4) circuit

For example, we can set unit values and "drive" them from inputs to outputs, using simple rules. If the input unit C is a logical unit ("ball"), then the signals installed at inputs A and B at the outputs are swapped. If there is a logical zero at input C (there is no ball), then the signals installed at inputs A and B go to the outputs without changes. Therefore, at the p=1, q=1, r=1 on outputs "parity" and "carry" are formed 1. This is forward mode. In the back mode we can install "parity"=1 and "carry"=1 and then "to roll" "balls" to the input p,q,r.

In this mode FG inputs and outputs are swapped: $F1$=C; $F2$=B; $F3$=A.

5 RESULTS

Based on the above studies, is proposed FG realization by1-LUT (Fig. 15).

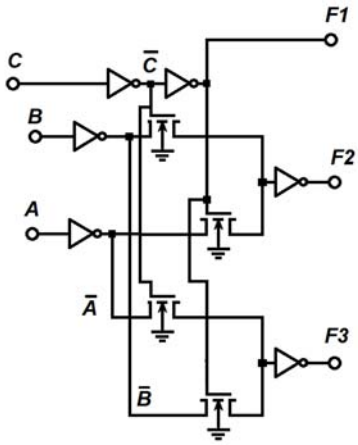


Figure 15 – Proposed FG realization by1-LUT

The signal C passes to the output $F1$ without changing, and two inverters play the role of a signal amplifier.

Signals B and A are transmitted to the outputs of either $F2$ or $F3$ depending on the value of signal C which controls the corresponding pass transistors.

1-LUT based FG simulation in Multisim CAD shown at Fig. 16.

© Tyurin S. F., 2020
DOI 10.15588/1607-3274-2020-1-5

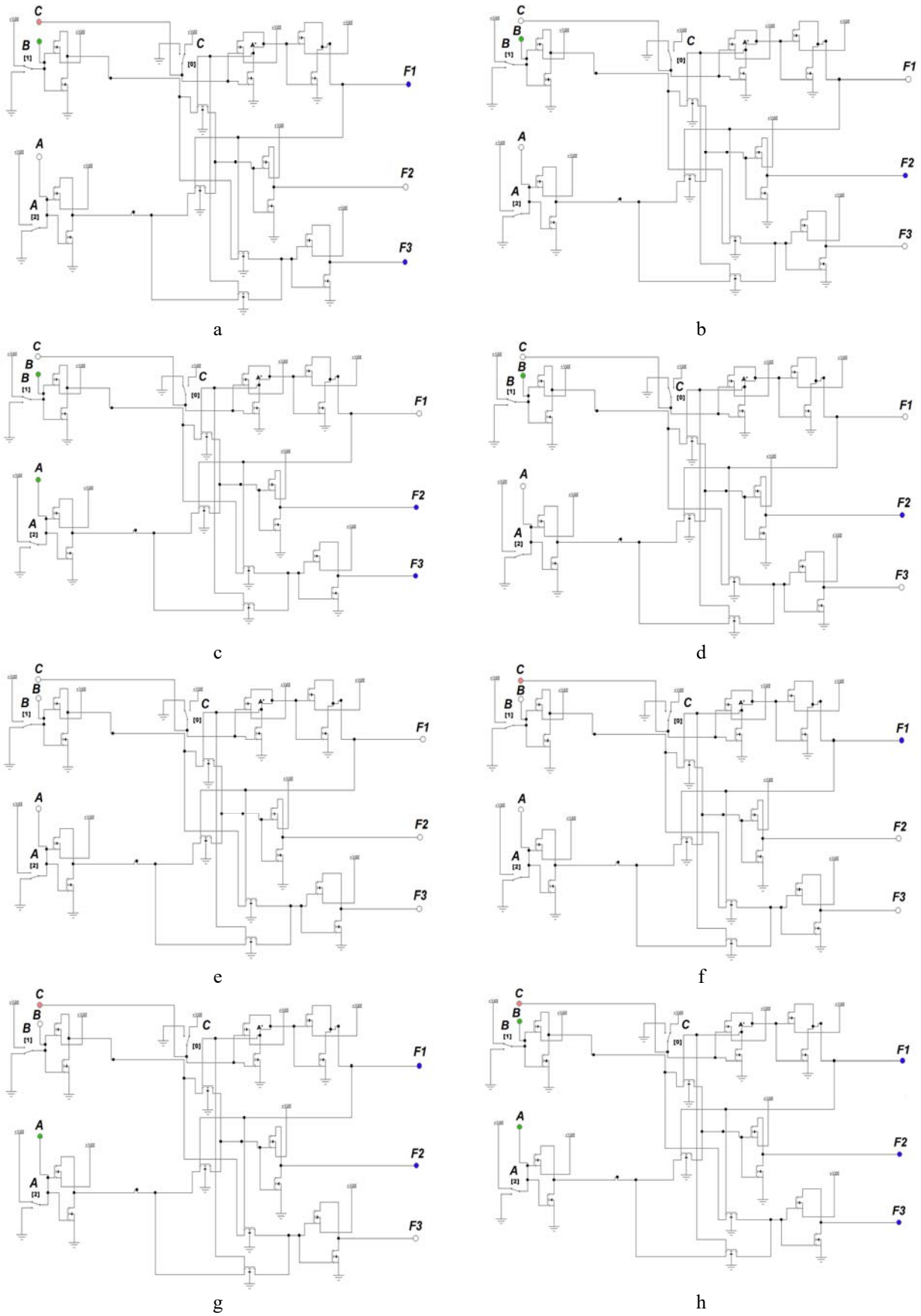


Figure 16 – 1-LUT based FG simulation in Multisim CAD:
a – C=1, B=1, A=0; b – C=0, B=1, A=0; c – C=0, B=1,A=1; d – C=0, B=1, A=0; e – C=0, B=0, A=0; f – C=1, B=0, A=0; g – C=1, B=0, A=1; h – C=1, B=1, A=1

© Tyurin S. F., 2020
DOI 10.15588/1607-3274-2020-1-5

6 DISCUSSION

Proposed LUT for one variable has two trees in different to usual LUT FPGA [18]. Two NOT gates on *C* input are needed to restore binary signal from another gates. Output *F*1 repeat control input *C*. Input *B* can be connected to the output *F*2 if *C*=0 or to the output *F*3 if *C*=1. Input *A* can be connected to the output *F*3 if *C*=0 or to the output *F*2 if *C*=1. NOT gates on input *B, C* and NOT gates on outputs *F*2, *F*3 are needed to restore binary signal from another gates too.

So there is bijective reflection *CBA* to *F*1*F*2*F*3 accordingly Truth Table 2. Such useful feature allows checking out gates in fault tolerant devices [19–21]. This implementation requires 16 transistors.

Successful simulation of the proposed FG-LUT shows Fig.16. We see at *C*=1, *B*=1, *A*=0 *F*1=1, *F*2=0, *F*3=1 (*B* and *A* swapped); at *C*=0, *B*=1, *A*=0 *F*1=0, *F*2=1, *F*3=0; at *C*=0, *B*=1, *A*=1 *F*1=0, *F*2=1, *F*3=1; at *C*=0, *B*=1, *A*=0 *F*1=0, *F*2=1, *F*3=0; at *C*=0, *B*=0, *A*=0 *F*1=0, *F*2=0, *F*3=0 etc. Table 2 proved.

As it evident from the Fig. 15,16 – this gate works in one mode "Forward" from the left to right. If we modify circuit Fig. 15 and include "Back" mode, we shall get Fig. 17.

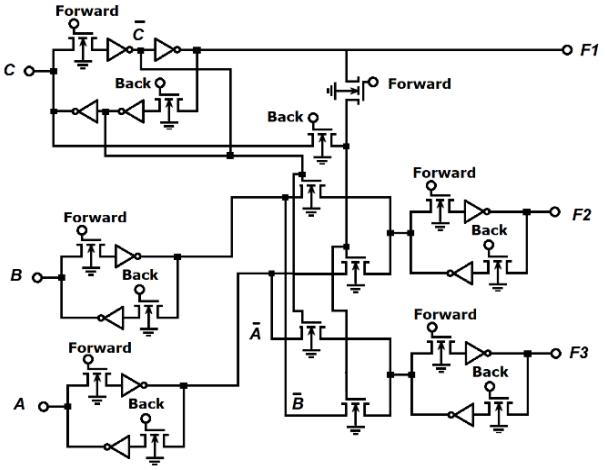


Figure 17 – Proposed 1-LUT based FG with two modes: Forward and Back

Complexity of modified gate is very larger: 24 (NOT gates) +16 (pass transistors) = 40 transistors. Therefore, we must find new ways two modes realization, for example by tri-states buffers; however, it increase time delay yet more and power consumption too.

Next discussion direction is using proposed gate in future CMOS adiabatic logic: how to control "balls" of power supply for the NOT gates? In addition, there is the problem of capacitors power supply leaking.

Creating of the fault tolerant reversible logic may be solve in view including redundancy [21–22] with considering restriction [23].

CONCLUSIONS

The problem of creating reversible logic gates and reversible circuits and devices is the point of growing modern IT. Boolean algebra allows to describe reversible logic analyses and synthesis, but complexity such math requires to design new more simply description in terms "swap" or "not swap" relate each variable.

The scientific novelty of obtained results is that the proposed synthesis method and proposed gates creates base for design reversible circuits.

The practical significance of obtained results lies in the fact that the simulation of the proposed gates confirmed their effectiveness, which allows you to create reversible systems in FPGAs.

Prospects for further research are to study the problem of checking out and diagnosis of gates. Optimization of the amount repeaters for the fan-out imitation is interest direction. Design layout of the proposed gate may be subject of the next article. Design software for automatic synthesis proposed FG circuits may be subject of the new science projects.

ACKNOWLEDGEMENTS

This research was carried out with the support of the Department of Automation and Remote Control of the Perm National Research Polytechnic University (Head of the department Prof. Yuzhakov Alexander Anatolievich) and of the Department of Software Computing Systems of the Perm State University.

Grate thanks to the PhD Oleg V. Goncharovskiy, Drs Vladimir I. Freiman and Irina A. Barinova (Perm National Research Polytechnic University) too.

Special thanks to the honored inventor of Ukraine, Doctor of Technical Sciences, Professor Kharchenko Vyacheslav Sergeyevich, phD Oleg Illiashenko (National Aerospace University "Kharkiv Aviation Institute") and to the Doctor of Technical Sciences, Professor Drozd Alexander Valentinovich (Odessa National Polytechnic University).

This work was supported in part by a grant from EU by TEMPUS-GreenCo (530270-TEMPUS-1-2012-1-UK-TEMPUS-JPCR).

REFERENCES

1. Quantum Gates – [Electronic resource]. Access mode: https://web.archive.org/web/20150104033422/http://web.stanford.edu/~rsasaki/AP226/AP226.html
2. Morita K. Reversible Logic Gates. [Electronic resource]. Access mode: https://link.springer.com/chapter/10.1007%2F978-4-431-56606-9_4
3. Reversible Computing. [Electronic resource]. Access mode: http://web.eecs.utk.edu/~bmaclenn/Classes/494-594-UC-F14/handouts/LNUC-II.C.3-9.pdf
4. Lala P. K., Parkerson J. P., Chakraborty P. Adder Designs using Reversible Logic Gates. [Electronic resource]. Access mode: https://pdfs.semanticscholar.org/f782/40586e5c74688c439784498d8bd31bab866c.pdf
5. Rangaraju H. G, Venugopal U., Muralidhara K. N., Raja K. B. Low Power Reversible Parallel Binary Adder/Subtractor. [Electronic resource]. Access mode: https://arxiv.org/ftp/arxiv/papers/1009/1009.6218.pdf
6. Xu S. Reversible Logic Synthesis with Minimal Usage of Ancilla Bits. [Electronic resource]. Access mode: https://arxiv.org/pdf/1506.03777.pdf
7. Mubin U. H., Zarrin T. S., Hafiz M. An Improved Design of a Reversible Fault Tolerant LUT-based FPGA, *29th International Conference on VLSI Design and 2016 15th International Conference on Embedded Systems*, VLSID-2016, 4–8 Jan. 2016: proceedings.

© Tyurin S. F., 2020
DOI 10.15588/1607-3274-2020-1-5

8. Fredkin E., Toffoli T. Conservative logic, International Journal of Theoretical. Physics, 21, 3/4, p. 219–253, 1982.
9. Patel1 R. B., Ho J., Ferreyrol1 F., Ralph T. C., Pryde G. J. A quantum Fredkin gate [Electronic resource]. Access mode: https://advances.sciencemag.org/content/2/3/e1501531
10. Rosanna M., Prasanna M., Amudha S. Implementation of testable reversible sequential circuit on FPGA, *2015 International Conference on Innovations in Information, Embedded and Communication Systems, ICIIECS–2015, 19–20 March 2015: proceedings*. Coimbatore, India, IEEE, 2015, pp. 145–150. DOI: 10.1109/ICIIECS.2015.7192888
11. Himanshu T., Vinod A. P. Design of Reversible Sequential Elements With Feasibility of Transistor Implementation, *2007 IEEE International Symposium on Circuits and Systems, 27–30 May 2007: proceedings*. New Orleans, LA, USA: IEEE, 2007, pp. 121–126. DOI: 10.1109/ISCAS.2007.378815
12. Yelekar R. P., Sujata S. C. Design of sequential circuit using reversible logic, *IEEE-International Conference On Advances In Engineering, Science And Management, ICAESM–2012, 30–31 March 2012: proceedings*. Nagapattinam, Tamil Nadu, India, IEEE, 2012, pp. 321–326
13. Kiran J., Binu K. M. Implementation of a FIR filter model using reversible Fredkin Gate, *2014 International Conference on Control, Instrumentation, Communication and Computational Technologies, ICCICCT–2014, 10–11 July 2014: proceedings*. Kanyakumari, India, IEEE, 2014, pp. 125–132. DOI: 10.1109/ICCICCT.2014.6993048
14. Dueck G. W. Maslov D., Miller D.M. Transformation-based synthesis of networks of Toffoli/Fredkin gates, *Canadian Conference on Electrical and Computer Engineering. Toward a Caring and Humane Technology, CCECE –2003, 4–7 May 2003: proceedings*. Montreal, Quebec, Canada, IEEE, 2003, pp. 211–214 DOI: 10.1109/CCECE.2003.1226380
15. Kiran D. K., Akalpita L. K. Implementation of 4-Tap FIR Filter Using Fredkin Gate, *International Journal of Engineering Science and Computing*, June, 2016, Volume 6, Issue No. 6, pp. 6504–6508. DOI 10.4010/2016.1563
16. Himanshu T., Ranganathan N., Kotiyal S. Design of Testable Reversible Sequential Circuits, *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, July 2013, Volume 21, Issue No 7, pp. 1201–1209. DOI: 10.1109/TVLSI.2012.2209688
17. Shannon C. E. The synthesis of two-terminal switching circuits [Electronic resource]. Access mode: https://ieeexplore.ieee.org/document/6771698
18. Tyurin S. F., Grekov A. V. Study of the multy input LUT complexity, *Radio Electronics, Computer Science, Control*, 2018, No. 1, pp. 14–21. DOI: 10.15588/1607-3274-2018-1-2
19. Tyurin S. F. Investigation of a Hybrid Redundancy in the Fault-Tolerant Systems, *Radio Electronics, Computer Science, Control*, 2019, No. 2, pp. 23–33. 10.15588/1607-3274-2019-2-3
20. Tyurin S. F. LUT's Sliding Backup, *IEEE transactions on device and materials reliability*, 2019, Published, MAR, Vol. 19, Issue 1, pp. 221–225. DOI: 10.1109/TDMR.2019.2898724
21. Carmichael C. Triple Module Redundancy Design Techniques for Virtex FPGAs [Electronic resource]. Access mode: https://www.xilinx.com/support/documentation/application_notes/xapp197.pdf
22. El-Maleh A. H., Al-Yamani A., Al-Hashimi B. M. Transistor-Level Defect Tolerant Digital System Design at the Nanoscale. Research Proposal Submitted to Internal Track Research Grant Programs [Electronic resource]. Access mode: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.474.3844&rep=rep1&type=pdf
23. Mead C. A., Conway L. Introduction to VLSI Systems. [Electronic resource]. Access mode: https://www.researchgate.net/publication/234388249_Introduction_to_VLSI_systems

Received 13.11.2019.
Accepted 24.02.2020.

УДК 004.93

ЕЛЕМЕНТ ФРЕДКІНА НА БАЗІ LUT

Тюрін С. Ф. – д-р техн. наук, професор, професор кафедри автоматики і телемеханіки Пермського національного дослідницького політехнічного університету, Пермь, Росія; професор кафедри математичного забезпечення обчислювальних систем Пермського державного національного дослідницького університету, Пермь, Росія.

АНОТАЦІЯ

Актуальність. Концепція існуючих комп'ютерів при досягненні нанорозмірів апаратних засобів практично вичерпала себе. Це також відноситься до обчислювальної потужності і пов'язаними з нею витратами на енергію. Оборотні обчислення, наприклад, більярдний комп'ютер, є базовою моделлю квантових обчислень, які вважаються перспективою IT-технологій. Більярдні обчислення – це енергозберігаючі або екологічно чисті обчислення. Основою такої парадигми є спеціальні логічні елементи. Однак, математичний апарат для створення таких комп'ютерів ще не повністю розроблений. Проблема полягає в тому, що для нових оборотних елементів, які мають взаємно однозначну відповідність між входами і виходами, застосування відомих методів аналізу і синтезу стикається з певними труднощами. Так, наприклад, заборонено використовувати розгалуження, що істотно ускладнює синтез. Оборотні елементи повинні забезпечувати передачу сигналу в прямому і зворотному напрямках, що в принципі можливо в двійковій логіці, заснованої на буферах з трьома станами, але значно ускладнює пристрій, збільшує площу кристала і енергоспоживання, які вони покликані зменшити.

Мета. Метою даної роботи є аналіз функціонально завершених оборотних вентилів, які названі вентилем Тоффолі, вентилем Фредкіна, аналіз довічного повного суматора, заснованого на вентилях Фредкіна, і запропонований логічний метод синтезу проектування на основі запропонованого логічного елемента.

Методи. Аналіз цифрових схем з використанням булевої алгебри. Синтез цифрових схем запропонованим методом декомпозиції. Дизайн вентиля Фредкіна на базі LUT FPGA. Моделювання запропонованого елемента в системі NI Multisim від National Instruments Electronics Workbench Group.

Результати. Аналіз повного суматора на основі вентилів Фредкіна. Метод синтезу оборотних схем на основі вентилів Фредкіна. Вентиль Фредкіна на основі LUT і його моделювання.

Висновки. Проведені дослідження дозволяють будувати схеми на основі вентилів Фредкіна із запропонованих нових елементів.

КЛЮЧОВІ СЛОВА: квантові обчислення, логічна функція, вентиль Фредкіна, розкладання Шеннона булева факторизация.

УДК 004.93

ЭЛЕМЕНТ ФРЕДКИНА НА БАЗЕ LUT

Тюрин С. Ф. – д-р техн. наук, профессор, профессор кафедры автоматики и телемеханики Пермского национального исследовательского политехнического университета, Пермь, Россия; профессор кафедры математического обеспечения вычислительных систем Пермского государственного национального исследовательского университета, Пермь, Россия.

© Tyurin S. F., 2020
DOI 10.15588/1607-3274-2020-1-5

АННОТАЦИЯ

Актуальность. Концепция существующих компьютеров при достижении наноразмеров аппаратных средств практически исчерпала себя. Это также относится к вычислительной мощности и связанными с ней затратами на энергию. Обратимые вычисления, например, бильярдный компьютер, являются базовой моделью квантовых вычислений, которые считаются перспективой ИТ-технологий. Бильярдные вычисления – это энергосберегающие или экологически чистые вычисления. Основой такой парадигмы являются специальные логические элементы. Однако математический аппарат для создания таких компьютеров еще не полностью разработан. Проблема заключается в том, что для новых обратимых элементов, которые имеют взаимно однозначное соответствие между входами и выходами, применение известных методов анализа и синтеза сталкивается с определенными трудностями. Так, например, запрещено использовать разветвление, что существенно усложняет синтез. Обратимые элементы должны обеспечивать передачу сигнала в прямом и обратном направлениях, что в принципе возможно в двоичной логике, основанной на буферах с тремя состояниями, но значительно усложняет устройство, увеличивает площадь кристалла и энергопотребление, которые они призваны уменьшить.

Цель. Анализ функционально завершенных обратимых вентилей, которые названы вентилем Тоффоли, вентилем Фредкина, анализ двоичного полного сумматора, основанного на вентилях Фредкина, и предложенный логический метод синтеза проектирования на основе предложенного логического элемента.

Методы. Анализ цифровых схем с использованием булевой алгебры. Синтез цифровых схем предложенным методом декомпозиции. Дизайн вентиля Фредкина на базе LUT FPGA. Моделирование предложенного элемента в системе NI Multisim от National Instruments Electronics Workbench Group.

Результаты. Анализ полного сумматора на основе вентилей Фредкина. Метод синтеза обратимых схем на основе вентилей Фредкина. Вентиль Фредкина на основе LUT и его моделирование.

Выводы. Проведенные исследования позволяют строить схемы на основе вентилей Фредкина из предложенных новых элементов.

КЛЮЧЕВЫЕ СЛОВА: квантовые вычисления, логическая функция, вентиль Фредкина, разложение Шеннона или булева факторизация.

ЛІТЕРАТУРА / ЛИТЕРАТУРА

1. Quantum Gates – [Electronic resource]. – Access mode: https://web.archive.org/web/20150104033422/http://web.stanford.edu/~rsasaki/AP226/AP226.html
2. Morita K. Reversible Logic Gates. [Electronic resource] / K. Morita. – Access mode: https://link.springer.com/chapter/10.1007%2F978-4-431-56606-9_4
3. Reversible Computing. [Electronic resource]. Access mode: http://web.eecs.utk.edu/~bmaclenn/Classes/494-594-UC-F14/handouts/LNUC-II.C.3-9.pdf
4. Lala P. K. Adder Designs using Reversible Logic Gates. – [Electronic resource] / P. K. Lala , J. P. Parkerson, P. Chakraborty. – Access mode: https://pdfs.semanticscholar.org/f782/40586e5c74688c439784498d8bd31bab866c.pdf
5. Low Power Reversible Parallel Binary Adder/Subtractor. [Electronic resource] / [H. G. Rangaraju, U. Venugopal, K. N. Muralidhara, K. B. Raja]. – Access mode: https://arxiv.org/ftp/arxiv/papers/1009/1009.6218.pdf
6. S. Xu. Reversible Logic Synthesis with Minimal Usage of Ancilla Bits. [Electronic resource]. Access mode: https://arxiv.org/pdf/1506.03777.pdf
7. Mubin U. H. An Improved Design of a Reversible Fault Tolerant LUT-based FPGA / U. H. Mubin, T. S. Zarrin, M. Hafiz // 29th International Conference on VLSI Design and 2016 15th International Conference on Embedded Systems, VLSID-2016, 4–8 Jan. 2016: proceedings.
8. Fredkin E. Conservative logic. / E. Fredkin, T. Toffoli // International Journal of Theoretical. – Physics. – 21, 3/4. – P. 219–253, 1982.
9. A quantum Fredkin gate [Electronic resource] / [R. B. Patel1, J. Ho, F. Ferreyrol1 et al.]. – Access mode: https://advances.sciencemag.org/content/2/3/e1501531
10. Rosanna M. Implementation of testable reversible sequential circuit on FPGA / M. Prasanna, S. Amudha // 2015 International Conference on Innovations in Information, Embedded and Communication Systems, ICIIECS–2015, 19–20 March 2015: proceedings. – Coimbatore, India: IEEE, 2015. – P. 145–150. DOI: 10.1109/ICIIECS.2015.7192888
11. Himanshu T. Design of Reversible Sequential Elements With Feasibility of Transistor Implementation / T. Himanshu, A. P. Vinod // 2007 IEEE International Symposium on Circuits and Systems, 27–3 0 May 2007: proceedings. – New Orleans, LA, USA: IEEE, 2007. – P. 121–126. DOI: 10.1109/ISCAS.2007.378815
12. Yelekar R. P. Design of sequential circuit using reversible logic / R. P. Yelekar, S. C. Sujata // IEEE-International Conference On Advances In Engineering, Science And Management, ICAESM – 2012, 30–31 March 2012: proceedings. – Nagapattinam, Tamil Nadu, India : IEEE, 2012. – P.321–326
13. Kiran J. Implementation of a FIR filter model using reversible Fredkin Gate / J. Kiran, K. M. Binu // 2014 International Conference on Control, Instrumentation, Communication and Computational Technologies, ICCICCT –2014, 10–11 July 2014: proceedings. – Kanyakumari, India: IEEE, 2014. – P. 125–132. DOI: 10.1109/ICCICCT.2014.6993048
14. Dueck G.W. Transformation-based synthesis of networks of Toffoli/Fredkin gates / G. W. Dueck, D. Maslov, D. M. Miller // Canadian Conference on Electrical and Computer Engineering. Toward a Caring and Humane Technology, CCECE –2003, 4–7 May 2003: proceedings. – Montreal, Quebec, Canada : IEEE, 2003. – P. 211–214. DOI: 10.1109/CCECE.2003.1226380
15. Kiran D. K. Implementation of 4-Tap FIR Filter Using Fredkin Gate / D. K. Kiran, L. K. Akalpita // International Journal of Engineering Science and Computing. – June 2016. – Volume 6, Issue No. 6. – P. 6504–6508. DOI 10.4010/2016.1563
16. Himanshu T. Design of Testable Reversible Sequential Circuits / T. Himanshu, N. Ranganathan, S. Kotiyal // IEEE Transactions on Very Large Scale Integration (VLSI). – 2013. – July Volume 21, Issue No. 7. – pp. 1201–1209. DOI: 10.1109/TVLSI.2012.2209688
17. Shannon C. E. The synthesis of two-terminal switching circuits [Electronic resource] / C. E. Shannon. – Access mode: https://ieeexplore.ieee.org/document/6771698
18. Tyurin S. F. Study of the multy input LUT complexity / S. F. Tyurin, A. V. Grekov // Radio Electronics, Computer Science, Control. – 2018. – № 1. – P. 14–21. DOI: 10.15588/1607-3274-2018-1-2
19. Tyurin S.F. Investigation of a Hybrid Redundancy in the Fault-Tolerant Systems // Radio Electronics, Computer Science, Control. – 2019. – № 2. – P. 23–33. 10.15588/1607-3274-2019-2-3
20. Tyurin S. F. LUT's Sliding Backup. IEEE transactions on device and materials reliability. – 2019. – Published: MAR. – Vol. 19, Issue 1. – P. 221–225. DOI: 10.1109/TDMR.2019.2898724
21. Carmichael C. Triple Module Redundancy Design Techniques for Virtex FPGAs [Electronic resource]. – Access mode: https://www.xilinx.com/support/documentation/application_notes/xapp197.pdf
22. El-Maleh A. H. Transistor-Level Defect Tolerant Digital System Design at the Nanoscale. Research Proposal Submitted to Internal Track Research Grant Programs [Electronic resource] / A. H. El-Maleh, A. Al-Yamani, B. M. Al-Hashimi. – Access mode: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.474.3844&rep=rep1&type=pdf
23. Mead C. A. Introduction to VLSI Systems / C. A. Mead, L. Conway. – [Electronic resource]. – Access mode: https://www.researchgate.net/publication/234388249_Introduction_to_VLSI_systems

© Tyurin S. F., 2020
DOI 10.15588/1607-3274-2020-1-5