3. Коростелев, А. А. Теоретические основы радиолокации / А. А. Коростелев, И. Ф. Клюев, Ю. А. Мельник; под ред. В. Е. Дулевича – 2-е изд. переработ. и доп. – М. : Сов. Радио, 1978. – 608 с.

4. Бронштейн, И. Н. Справочник по математике. Для инженеров и учащихся ВТУЗОВ / И. Н. Бронштейн, К. А. Семендяев. – М. : Наука, 1980. – 976 с.

UDC 004.4

Inference of reversible decision diagrams for modelling adders

Prihozhy A. A.
Belarusian National Technical University, Minsk, Belarus

Abstract:
The quantum logical circuits are reversible since they are defined through using Boolean conjunction, exclusive-or and constants. We propose a technique of inferring ternary reversible decision diagrams at the aim of modelling and synthesis of quantum parallel many-bit adders.

The binary and ternary decision diagrams [1] are a means for modelling and synthesis of logical circuits. The theory of ternary diagrams is based on incompletely specified logical functions [2, 3]. The paper proposes a technique for inferring reversible ternary decision diagrams representing logical circuits of adders [4] for quantum implementation [5].

Let $x = (x_1,\ldots,x_n)$ be a vector of n scalar Boolean variables. A scalar Boolean function $f(x)$ is a mapping $B^n \to B$, $B = \{0, 1\}$. Let a Boolean vector function $F(x) = (f_1,\ldots,f_n)$: $B^n \to B^n$ is given by vector $(x_1,\ldots,x_{i-1}, f(x_1,\ldots,x_n), x_{i+1},\ldots,x_n)$ of scalar functions $f_1=x_1,\ldots, f_{i-1} = x_{i-1}, f_i = f, f_{i+1} = x_{i+1},\ldots, f_n = x_n$. In $F(x)$, the number of components is the same as the number of primary variables in $f(x)$. The Boolean function $f(x_1,\ldots,x_n)$ is n-reversible if there is an index $i \in \{1,\ldots,n\}$ for which the vector function $F(x) = (x_1,\ldots,x_{i-1}, f(x_1,\ldots,x_n), x_{i+1},\ldots,x_n)$ is bijective. All logical quantum circuits are reversible in nature and are represented by reversible functions.

It is shown in previous works that the binary Boolean *exclusive-or* operation which is given by $f = x_1 \oplus x_2$ is 2-reversible. The binary Boolean

equivalence operation $f = x_1 \equiv x_2$ being a *negation* $\neg$ of the exclusive-or is also 2-reversible. The ternary Boolean *xor-and-accumulation* operation $xac = x_1 \oplus (x_2 \wedge x_3)$ is 3-reversible. It is described through the binary *exclusive-or* and *conjunction* operations. At the same time, the binary Boolean *conjunction* $f = x_1 \wedge x_2$ and *disjunction* $f = x_1 \vee x_2$ need an ancilla and are therefore not 2-reversible. They are 3-reversible. Consequently, the binary Boolean *implication* $f = x_1 \rightarrow x_2$, *Sheffer stroke* $f = x_1 \mid x_2$ and *Pierce arrow* $f = x_1 \downarrow x_2$ are 3-reversible as well. The ternary Boolean *if-then-else* operation $ite = x_1 \wedge x_2 \vee \neg x_1 \wedge x_3$ ($\neg$ is Boolean negation) and the *majority* operation $maj = x_1 \wedge x_2 \vee x_1 \wedge x_3 \vee x_2 \wedge x_3$ are not 3-reversible. They demand one ancilla and therefore are 4-reversible. Any reversible Boolean function can be described by a superposition of the exclusive-or, xor-and-accumulation, constant 1 and constant 0 operations with the same or increased number of input and output variables (ancillas). If no ancillas are needed, we call the function reversible. Usually, optimization is needed for a given function to obtain a minimum number of ancillas.

In the widely used Binary Decision Diagrams (BDDs), the semantics of a nonterminal vertex labeled by a Boolean variable is defined by the Shannon expansion ($sha = x_i \wedge g \vee \neg x_i \wedge h$) using the ternary *ite* operation which is not 3-reversible. Each nonterminal vertex of the diagram has two outgoing edges. In the Binary Functional Decision Diagrams (BFDD), the semantics is defined by the positive ($davp = h \oplus x_i \wedge g$) or negative $davn = g \oplus \neg x_i \wedge h$) Davio expansion using the xor-and-accumulation operation which is 3-reversible. Therefore, BFDD is a reversible diagram whose nonterminal vertices use no ancillas. At the same time, BFDD has poor capabilities for modeling such digital devices as many-bit adders.

In the Ternary Decision Diagrams (TDDs), proposed in [1] as if-decision diagrams (IFDs), each nonterminal vertex has three outgoing edges. The theory of incompletely specified functions is used for creating the diagrams. The semantics of the nonterminal vertex is defined by the expansion $prih = d \wedge g \vee \neg d \wedge h$ where d, g and h are arbitrary Boolean functions of the same set of variables. In Positive Ternary Reversible Decision Diagrams (PTRDD), the semantics is defined by the expansion $prip = h \oplus d \wedge g$. PTRDD is a reversible diagram. It is suitable for modelling many-bit adders.

TDD (also known as IFD) is suitable for developers who specify logical (digital) circuits at the functional level. In contrast, it is not easy for

designers to describe a digital circuit using TRDD directly. At the same time, this diagram supports efficiently the circuit synthesis flow and can yield logical circuits of good quality. Especially, they can be successfully used in such applications as quantum computers design since all quantum logical circuits are reversible. The goal of the paper is to develop an effective technique of transforming TDD to TRDD.

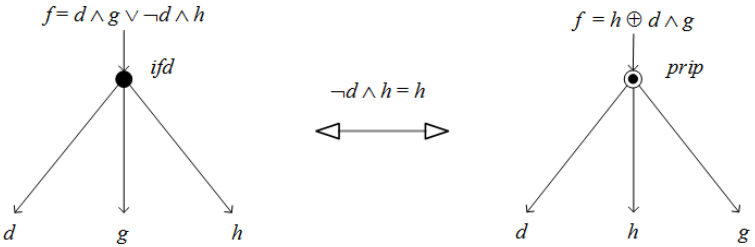

Figure 1 – *Rule* of transforming TDD's nonterminal vertex to PTRDD's nonterminal vertex and vice versa under condition $\neg d \wedge h = h$

Figure 1 depicts *Rule* of mapping a nonterminal vertex of TDD representing Boolean function $f = d \wedge g \vee \neg d \wedge h$ in the basis of $\{\neg, \wedge, \vee\}$ to a nonterminal vertex of PTRDD representing the same function in the basis of $\{\wedge, \oplus\}$. *Rule* can be used if $\neg d \wedge h = h$. The TDD's vertex has three outgoing edges directed to sub-diagrams *d*, *h* and *g*. *Rule* can be proved as

$$f = d \wedge g \vee \neg d \wedge h = d \wedge g \oplus \neg d \wedge h = d \wedge g \oplus h = h \oplus d \wedge g.$$

Let consider the 1-bit full adder and represent it with TDDs and TRDDs. Figure 2 (left) depicts a 2-root TDD for two functions of sum $s_i = a_i \oplus b_i \oplus c_{i-1}$ and carry $c_i = majority(a_i, b_i, c_{i-1})$ depending on inputs a_i, b_i and c_{i-1}.

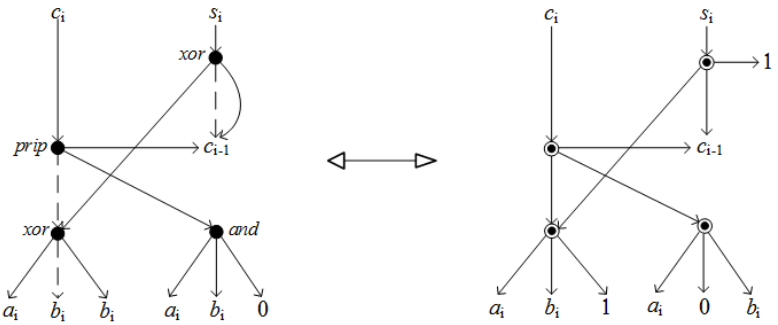

Figure 2 – Transforming TDD-representation (left) *prip* of one-bit full adder to PTRDD-representation (right) and vice versa (dashed line represents a complemented edge)

The diagram contains seven terminal vertices labeled by Boolean variables and constants, and four nonterminal vertices connected by twelve directed edges. The nonterminal vertices represent one *and*-operation, two *xor*-operations and one *prip* operation. Three of twelve edges are complemented, they represent the Boolean negation ¬ operation. Figure 2 (right) depicts a 2-root TRDD for the s_i and c_i functions. Each vertex of the TRDD corresponds to exactly one vertex of TDD and vice versa.

The vertices of TDD which are labeled with *xor* and *and* are mapped to TRDD's vertices as shown in Figure 3a and 3b respectively. The vertex labeled with *prip* describes the function

$$d \wedge g \vee \neg d \wedge h = g \oplus \neg d \wedge h$$

because

$d = \neg(a_i \oplus b_i)$, $g = a_i \wedge b_i$ and $h = c_{i-1}$

and

$$d \wedge g = \neg(a_i \oplus b_i) \wedge (a_i \wedge b_i) = (a_i \equiv b_i) \wedge (a_i \wedge b_i) = (a_i \wedge b_i) = g.$$

a) b)

Figure 3 – TRDD-representation of a) Boolean conjunction $\wedge$ and b) Boolean exclusive-or $\oplus$

The 3-bit carry ripple adder is constructed as a chain of 1-bit full adders. It is represented either with diagram $D1$ of type TDD (Figure 4) or with diagram $RD1$ of type TRDD (Figure 5). The two diagrams are generated independently of each other: the first one is not reversible, and the second one is reversible. The depth of both diagrams is the same and is equal to 4 nonterminal vertices.

For the many-bit adders represented with TDDs, work [4] proposed a method of parallelization that is based on cutting long chains of diagrams' vertices into shorter chains. Since the exclusive-or operation models the incompletely specified Boolean functions poorly, the method cannot be immediately applied to the reversible diagrams PTRDDs.

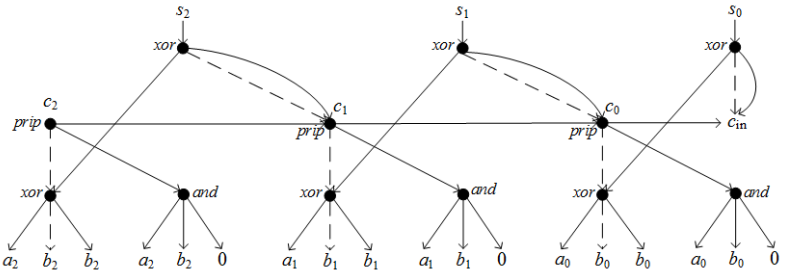

Figure 4 –TDD-representation $D1$ of 3-bit ripple-carry adder of depth 4

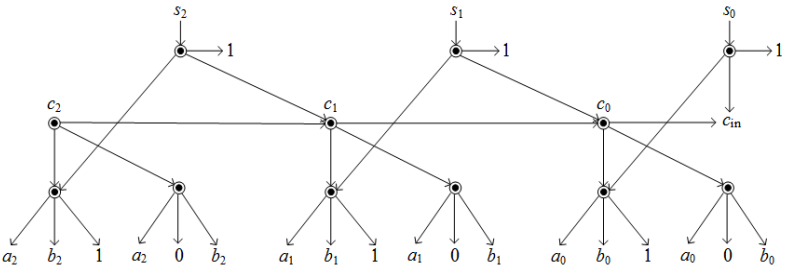

Figure 5 –PTRDD-representation $RD1$ of 3-bit ripple-carry adder

We propose a two-step technique which firstly cuts long chains in TDDs and secondly maps the TDDs to PTRDDs using *Rule* shown in Figure 1. Let demonstrate the technique at the aim of parallelization of the carry part of the 3-bit ripple carry adder (Figure 4). We break the chain of diagram $D1$ consisting of four nonterminal vertices labeled *prip*, *prip*, *prip* and *and*, and obtain a TDD-representation $D2$ depicted in Figure 6. The resulting 3-root TDD has two additional nonterminal vertices z_0 and z_1 of type *prip* and *and* respectively. The longest path of $D2$ consists of three nonterminal vertices instead of four in $D1$.

At the second step the procedure maps the diagram $D2$ to a reversible diagram $RD2$ depicted in Figure 7. Let us prove that $D2$ and $RD2$ describe the same behavior, i.e., they are functionally equivalent. All vertices of $D2$ of type *xor* and *and* are mapped to functionally equivalent reversible vertices shown in Figure 3. Three vertices c_0, c_1 and z_1 have the type *prip* since they meet the condition $\neg d \wedge h = h$ of *Rule*.

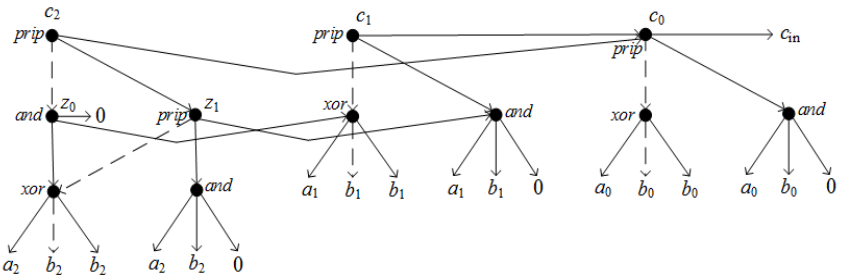

Figure 6 – TDD-representation $D2$ of parallel (3-depth) carry part of 3-bit adder prip

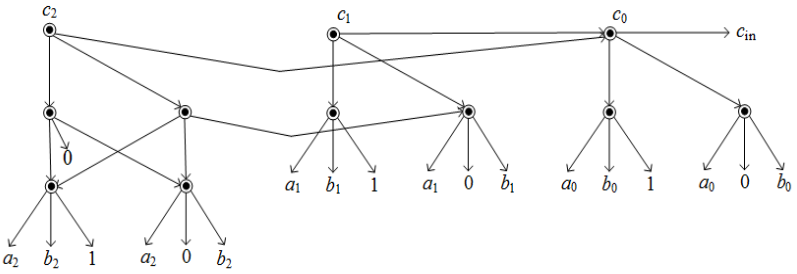

Figure 7 – PTRDD-representation *RD*2 of parallel (3-depth) carry part of 3-bit adder

Let us prove that vertex c_2 also has type *prip*. The vertex's semantics is defined as

$$c_2 = d \wedge g \vee \neg d \wedge h = g \oplus \neg d \wedge h$$

since

$$d = \neg z_0 = \neg((a_2 \oplus b_2) \wedge (a_1 \oplus b_1)) = (a_2 \equiv b_2) \vee (a_1 \equiv b_1),$$
$$g = z_1 = (a_2 \wedge b_2) \vee ((a_1 \wedge b_1) \wedge (a_2 \oplus b_2)),$$
$$h = c_0,$$

and

$$\begin{aligned} d \wedge g &= ((a_2 \equiv b_2) \vee (a_1 \equiv b_1)) \wedge ((a_2 \wedge b_2) \vee ((a_1 \wedge b_1) \wedge (a_2 \oplus b_2))) = \\ &= ((a_2 \equiv b_2) \wedge (a_2 \wedge b_2)) \vee ((a_2 \equiv b_2) \wedge ((a_1 \wedge b_1) \wedge (a_2 \oplus b_2))) \vee \\ &\quad ((a_1 \equiv b_1) \wedge (a_2 \wedge b_2)) \vee ((a_1 \equiv b_1) \wedge ((a_1 \wedge b_1) \wedge (a_2 \oplus b_2))) = \\ &= (a_2 \wedge b_2) \vee ((a_1 \equiv b_1) \wedge (a_2 \wedge b_2)) \vee ((a_1 \wedge b_1) \wedge (a_2 \oplus b_2)) = \\ &= (a_2 \wedge b_2) \vee ((a_1 \wedge b_1) \wedge (a_2 \oplus b_2)) = g. \end{aligned}$$

Therefore, vertex c_2 has type *prip*.

Conclusion. The problem of modelling digital circuits and the aim of implementing adders is considered. In the paper, the adders are modeled by ternary decision diagrams also known as if-decision diagrams. The diagrams are defined through using Boolean negation, conjunction, disjunction and constants. The quantum logical circuits are reversible, they are defined through using Boolean conjunction, exclusive-or and constants. We have proposed a technique of transforming ternary decision diagrams describing the adders to ternary reversible decision diagrams of the same size. The technique can be used for inferring parallel reversible ternary decision diagrams at the aim of synthesis of parallel quantum adders.

References

1. Prihozhy, A. A. If-Diagrams: Theory and Application. Proc. 7th Int. Workshop PATMOS'97. – UCL, Belgium, 1997. – P. 369 – 378.

2. Прихожий, А.А. Обобщение разложения Шеннона для частично определенных функций: теория и применение. Системный анализ и прикладная информатика. – 2013, № 1–2. – С. 6–11.

3. Прихожий, А. А. Новые разложения булевых функций по операции исключающее или в системах логического проектирования. Системный анализ и прикладная информатика. – 2014, № 1-3. – С. 9–16.

4. Prihozhy, A. A. Synthesis of parallel adders from if-decision diagrams. System Analysis and Applied Information Science, 2020, No 2, pp. 61–70.

5. Prihozhy, A. A. Synthesis of quantum circuits based on incompletely specified functions and if-decision diagrams, Journal of the Belarusian State University. Mathematics and Informatics, 2021, Volume 3, pp. 84–97.

УДК 621.521

Выбор откачных средств для вакуумных камер большого объема

Савчук Д. О., магистрант
Белорусский национальный технический университет
г. Минск, Республика Беларусь
Научный руководитель: к. ф.-м. н., доцент Босяков М. Н.

Аннотация:
В данной статье приводится методика расчета и подбора откачных средств для крупногабаритных рабочих камер установок ионного азотирования.

Вакуумные камеры имеют широкое применение в различных отраслях промышленности, поэтому для обеспечения их стабильной работы важен правильный выбор откачных средств, которые напрямую влияют на производительность.

Особенности вакуумных камер большого объема заключаются в необходимости быстрого и эффективного удаления значительных объемов воздуха и технологических газов. Это требует применения таких систем откачки, которые будут способны обеспечить требуемый уровень вакуума в минимальные сроки при максимальной энергоэффективности.