SOC-ОРИЕНТИРОВАННЫЕ ТЕХНОЛОГИИ МОЗГОПОДОБНЫХ КВАНТОВЫХ ВЫЧИСЛЕНИЙ

Владимир Хаханов, Murad Ali Abbas, Евгения Литвинова
Харьковский национальный университет радиоэлектроники, Украина
hahanov@kture.kharkov.ua

Abstract. *Qubit (quantum) structures of data and computational processes for significantly improving performance when solving problems of discrete optimization and fault-tolerant design are proposed. We describe a hardware-focused models for parallel (one cycle) calculating the power set (the set of all subsets) on the universe of n primitives for solving coverage problems, minimization of Boolean functions, data compression, analysis and synthesis of digital systems through the implementation of the processor structure in the form of the Hasse diagram.*

Ключевые слова: *квантовые вычисления, кубитный процессор, диаграмма Хассе.*

I. Введение

Квантовые вычисления в последние годы становятся интересными для анализа кибернетического пространства, создания новых Интернет технологий, благодаря их некоторой альтернативности существующим моделям вычислительных процессов. Кроме того, рыночная привлекательность квантовых или кубитных моделей основывается на высоком параллелизме решения практически всех задач дискретной оптимизации, факторизации, минимизации булевых функций, эффективного сжатия, компактного представления и телепортации данных, отказоустойчивого проектирования [1-10] за счет существенного повышения аппаратных затрат. Но такая плата в настоящее время вполне допустима, поскольку существуют проблемы заполнения площадей силиконового кристалла, который содержит до 1 миллиарда вентилей при толщине пластины, равной 5 микрон. При этом современные технологии допускают создание пакета или «сэндвича», содержащего до 7 кристаллов, что соизмеримо с объемом нейронов головного мозга человека. Практически «беспроводное» соединение таких пластин основывается на технологической возможности сверления порядка 10 тысяч сквозных отверстий (vias) на 1 квадратном сантиметре. Наполнить полезной функциональностью такой объем допустимых на кристалле вентилей в настоящее время проблематично. Поэтому можно и нужно использовать «жадные» к аппаратуре модели и методы для создания быстродействующих средств параллельного решения практических задач. Имея ввиду дискретность и многозначность алфавитов описания информационных процессов, свойство параллелизма, заложенное в квантовых вычислениях, является особенно востребованным при создании эффективных и интеллектуальных «движков» для киберпространства или Интернета [11]; средств синтеза отказоустойчивых цифровых примитивов и систем [12]; тестирования и моделирования цифровых систем на кристаллах [13-15]; технологий защиты информации и компьютерных систем [5-7]. Здесь не рассматривается физическая основа квантовых вычислений, изначально заложенная в трудах ученых, ориентированных на возможность использования недетерминированных квантовых взаимодействий на уровне атома [16-17].

II. Кубитные, квантовые модели данных и вычислительных процессов

Квантовый компьютер предназначен для отказоустойчивого проектирования и решения оптимизационных задач, связанных с полным перебором на основе использования теории множеств. Особенность в том, что множество элементов в компьютере все равно упорядоченно, поскольку каждый байт имеет свой адрес. Поэтому теоретико-множественные операции сводятся к перебору всех адресов примитивных элементов. Адресный порядок структур данных хорош для задач, где компоненты моделей можно строго ранжировать, что дает возможность выполнять их анализ за один проход или одну итерацию. Там, где нет порядка в структуре, например, множество всех подмножеств, классическая модель памяти и вычислительных процессов наносит вред времени анализа ассоциации равных по рангу примитивов, или, в лучшем случае, обработка ассоциативных групп является неэффективной. Что можно предложить для неупорядоченных данных вместо строгого порядка? Процессор, где элементарной ячейкой служит образ или шаблон универсума из n примитивов, который генерирует $Q = 2^n$ всех возможных состояний такой ячейки в виде булеана или множества всех подмножеств. Прямое решение создания такой ячейки основано на унитарном позиционном кодировании состояний примитивов, которое с помощью суперпозиции последних образует множество всех подмножеств, формирующее в пределе универсум примитивов. Например, четыре примитива создают булеан, содержащий шестнадцать состояний (сочетаний), с помощью четырех двоичных разрядов:

A={Q=(1000), E=(0100), H=(0010), J=(0001), O={Q,H}=(1010), I={E,J}=(0101), A={Q,E}=(1100), B={H,J}=(0011), S={Q,J}=(1001, P={E,H}=(0110), C={E,H,J}=(1110), F={Q,H,J}=(1011), L={Q,E,J}=(1101), V={Q,E,H}=(1110), Y={Q,E,H,J} =(1111), U=(0000)}.

Операции над символами теоретико-множественного алфавита сводятся к логическим командам and, or, not, xor, которые формируют функционально полный базис, согласно теореме Поста. Например, ниже представлены логические преобразования теоретико-множественных операций:

$$Q \cup E = 1000 \vee 0100 = 1100 = A;$$
$$S \cap V = 1001 \wedge 1110 = 1000 = Q;$$
$$\tilde{B} = \overline{0011} = 1100 = A;$$
$$F \Delta P = 1011 \oplus 0110 = 1101 = Y;$$
$$H \Delta J = 0010 \oplus 0001 = 0011 = B;$$
$$F \Delta Y = 1011 \oplus 1111 = 0100 = Y;$$
$$F \Delta F = 1011 \oplus 1011 = 0000 = U(\varnothing);$$

Другая интерпретация булеана из четырех примитивов (двоичные коды: 00, 01, 10, 11), представленная ниже:

00	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
01	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
10	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
11	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

создает 16 различных функций от двух переменных. В то же время последнюю таблицу можно представить как коды символов многозначного алфавита, которыми легко оперировать для решения задач синтеза и анализа булевых функций:

Q	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
E	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
H	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
J	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
	∅	J	H	B	E	I	P	C	Q	S	O	F	A	L	V	Y

Такую таблицу легко построить для любого числа примитивов (n=2, 3,4, 5, 6, 7, 8 …), где теоретико-множественные операции над символами сводятся к логическим операциям над векторами. Два наиболее традиционных примитива (0,1) создают известный алфавит Кантора:

0	0 0 1 1
1	0 1 0 1
	∅ 1 0 X

Многозначность символов алфавита положительно влияет на минимизацию булевых функций. Например, компактное представление состояния входных переменных отдельных булевых функций от двух переменных при их кодировании символами 16-значного алфавита имеет не более двух кубов многозначного покрытия:

$$\begin{array}{|c|c|}\hline 00 & 0\\ 01 & 0\\ 10 & 0\\ 11 & 0\\ \hline\end{array} = \begin{array}{|c|c|}\hline Q & 0\\ E & 0\\ H & 0\\ J & 0\\ \hline\end{array} = \begin{array}{|c|c|}\hline Y & 0\\ \hline\end{array} = \begin{array}{|c|c|}\hline 1111 & 0\\ \hline\end{array}; \quad \begin{array}{|c|c|}\hline 00 & 0\\ 01 & 0\\ 10 & 0\\ 11 & 1\\ \hline\end{array} = \begin{array}{|c|c|}\hline Q & 0\\ E & 0\\ H & 0\\ J & 1\\ \hline\end{array} = \begin{array}{|c|c|}\hline V & 0\\ J & 1\\ \hline\end{array} = \begin{array}{|c|c|}\hline 1110 & 0\\ 0001 & 1\\ \hline\end{array};$$

$$\begin{array}{|c|c|}\hline 00 & 0\\ 01 & 0\\ 10 & 1\\ 11 & 0\\ \hline\end{array} = \begin{array}{|c|c|}\hline Q & 0\\ E & 0\\ H & 1\\ J & 0\\ \hline\end{array} = \begin{array}{|c|c|}\hline F & 0\\ H & 1\\ \hline\end{array} = \begin{array}{|c|c|}\hline 1101 & 0\\ 0010 & 1\\ \hline\end{array}; \quad \begin{array}{|c|c|}\hline 00 & 0\\ 01 & 0\\ 10 & 1\\ 11 & 1\\ \hline\end{array} = \begin{array}{|c|c|}\hline Q & 0\\ E & 0\\ H & 1\\ J & 1\\ \hline\end{array} = \begin{array}{|c|c|}\hline A & 0\\ B & 1\\ \hline\end{array} = \begin{array}{|c|c|}\hline 1100 & 0\\ 0011 & 1\\ \hline\end{array};$$

$$\begin{array}{|c|c|}\hline 00 & 1\\ 01 & 1\\ 10 & 1\\ 11 & 1\\ \hline\end{array} = \begin{array}{|c|c|}\hline Q & 1\\ E & 1\\ H & 1\\ J & 1\\ \hline\end{array} = \begin{array}{|c|c|}\hline Y & 1\\ \hline\end{array} = \begin{array}{|c|c|}\hline 1111 & 1\\ \hline\end{array}; \quad \begin{array}{|c|c|}\hline 00 & 0\\ 01 & 0\\ 10 & 1\\ 11 & 0\\ \hline\end{array} = \begin{array}{|c|c|}\hline Q & 0\\ E & 0\\ H & 1\\ J & 0\\ \hline\end{array} = \begin{array}{|c|c|}\hline F & 0\\ H & 1\\ \hline\end{array} = \begin{array}{|c|c|}\hline 1101 & 0\\ 0010 & 1\\ \hline\end{array};$$

$$\begin{array}{|c|c|}\hline 00 & 0\\ 01 & 1\\ 10 & 1\\ 11 & 1\\ \hline\end{array} = \begin{array}{|c|c|}\hline Q & 0\\ E & 1\\ H & 1\\ J & 1\\ \hline\end{array} = \begin{array}{|c|c|}\hline Q & 0\\ C & 1\\ \hline\end{array} = \begin{array}{|c|c|}\hline 1000 & 0\\ 0111 & 1\\ \hline\end{array}; \quad \begin{array}{|c|c|}\hline 00 & 0\\ 01 & 1\\ 10 & 1\\ 11 & 0\\ \hline\end{array} = \begin{array}{|c|c|}\hline Q & 0\\ E & 1\\ H & 1\\ J & 0\\ \hline\end{array} = \begin{array}{|c|c|}\hline S & 0\\ P & 1\\ \hline\end{array} = \begin{array}{|c|c|}\hline 1001 & 0\\ 0110 & 1\\ \hline\end{array}.$$

Таким образом, переход от двоичных векторов входных сигналов к символам замкнутого многозначного алфавита дает принципиально новую возможность к минимизации кубических покрытий (таблиц истинности), которые всегда будут иметь не более двух кубов $f(X)=\{C_1, C_0\}$, $C_1 \cup C_0 = U$, $C_1 = \overline{C}_0$, формирующих единичное и нулевое значение выхода функции двумя взаимно дополняющими символами, в совокупности формирующими символ-универсум. При этом мощность универсума примитивов $\mathrm{card}U = n$ формирует общее число состояний $Q = 2^n$ или производных от них символов, которые кодируются 2^n двоичными разрядами. Последующее двоичное кодирование входного символа каждого из двух кубов дает возможность максимально приблизиться к реализации функционального примитива как элемента памяти программируемых логических устройств (PLD), где входное слово логического элемента есть адрес ячейки памяти (бита), в котором записано состояние выхода. Однако таблица истинности в форме памяти есть ДНФ, которая необратима для решения задачи обратной импликации. Здесь выходом может служить явное задание функциональности в форме кубического покрытия, а точнее двух кубов покрытия, задающих все возможные решения по входам. При этом все логические элементы становятся одновходовыми, где входом является регистровая переменная или n-разрядный вектор, формирующий адрес памяти, хранящей $Q = 2^n$ бит, как значений функции $Y = f(A) = f(x_1, x_2, ..., x_i, ..., x_n)$.

Кубит есть двоичный вектор, содержащий n битов, для задания булеана (множества всех подмножеств) состояний $Q = 2^n$ на основе использования n примитивных символов (элементов).

Кубит – совокупность равнозначных двоичных n битов, формирующих единичным значением n примитивов, для обозначения $Q = 2^n$ состояний, составляющих булеан – множество всех подмножеств от n примитивов.

Здесь нет чисел! Все биты кубита равны при создании примитивов и отличаются только адресом. Любая теоретико-множественная операция выполняется за один такт, что невозможно при задании ассоциации примитивов на счетном (упорядоченном) пространстве памяти компьютера. Метрика (векторная и скалярная) анализа расстояний, предложенная в [11,13], абсолютно пригодная для измерения взаимодействия многозначных (двоичных) кубитных объектов, процессов и явлений, путем использования xor-операции.

В идеале использование кубитной структуры дает возможность представить любую функциональность в виде двух кубов, привязанных к нулю и единице. Такие кубы формируют КНФ и ДНФ соответственно. Можно упрощать и далее путем исключения из рассмотрения нуля и единицы, неявно имея их в виду. При этом два куба, формирующие входные условия, будут всегда взаимно инверсными, поскольку они дополняют друг друга до универсума примитивов. Следовательно, необходимо оставлять лишь одну букву (символ), а значит один двоичный код, который есть таблица истинности (двухвходового) функционального примитива:

$$\begin{array}{|c|c|}\hline 00 & 0 \\ 01 & 1 \\ 10 & 1 \\ 11 & 0 \\ \hline\end{array} = \begin{array}{|c|c|}\hline Q & 0 \\ E & 1 \\ H & 1 \\ J & 0 \\ \hline\end{array} = \begin{array}{|c|c|}\hline S & 0 \\ P & 1 \\ \hline\end{array} = \begin{array}{|c|c|}\hline 1001 & 0 \\ 0110 & 1 \\ \hline\end{array} \to P = \boxed{0110}$$

$$Y = P = E \vee H = A_1 \vee A_2 = \bar{x}_1 x_2 \vee x_1 \bar{x}_2.$$

Полученный вектор интерпретируется здесь не только как совокупность адресуемых битов, но как куб, формирующий единичное значение выхода примитива, над которым можно выполнять параллельные векторные логические операции. Иначе, совокупность или множество входных векторов можно трансформировать к одному вектору, где каждая позиция (адрес) соответствует входной комбинации, а значение позиции – состоянию выхода функции. Это дает возможность уменьшить экспоненциальную (2^n) вычислительную сложность синтеза кубического покрытия функциональности на основе заданных таблиц истинности структурных компонентов цифрового устройства до линейной функции за счет увеличения числа битов при задании переменных с n до 2^n разрядов.

Таким образом, структура куба функциональности – вектора выходных значений функционального примитива – изоморфна ДНФ, которая оперирует единичными термами булевых переменных. Аналогично можно записать КНФ, используя нулевой куб покрытия функционального примитива.

Результат минимизации интересен и для сжатия информации путем получения минимальной ДНФ, с помощью которой всегда можно восстановить исходную таблицу истинности. Вычислительная сложность минимизации в кубитном исчислении на многозначном алфавите линейна относительно числа переменных. Возникает естественный вопрос. Зачем минимизировать функцию, если она компактно представляется состояниями выходов по всем адресам, составленным из кодов от любого числа переменных. Единственная практически ориентированная цель – выполнение процедур обратной импликации для реализации регулярных алгоритмов синтеза тестов, которые на порядок улучшат свое быстродействие их выполнения за счет минимального числа кубов в покрытиях функциональных элементов.

Таким образом, на основании введенных кубитных структур данных и Хассе-модели вычислительного процесса можно сделать некоторые выводы:

Quantum Computer создавали специалисты в области квантовой механики, которые привнесли идею создания нечислового компьютера на аналоговой основе представления информации.

Введенное понятие кубита соответствует булеану примитивов, что является идеальной нечисленной формой описания компонентов объекта для их анализа, синтеза и оптимизации дискретных объектов.

Формы представления кубита: 1. Символы универсума примитивов, генерирующие множество всех подмножеств (булеан); 2. Двоичные векторы, где булеан представляется сочетанием единичных значений; 3. Диаграмма Хассе, формирующая булеан всех возможных решений на графе; 4. Полный граф переходов, определяющий множество всех подмножеств переходов в виде дуг графа; 5. Геометрическое представление на плоскости кубита в виде точек и отрезков, соответствующих булеану.

На практике более 90% всех задач IT-индустрии, связанной с поиском информации в киберпространстве, ее распознаванием и принятием решений относится к области дискретной математики, где трудно найти место числовой арифметике.

Необходимо создавать ассоциативные логические мозгоподобные параллельные (квантовые) процессоры, эффективно оперирующие примитивами булеана (кубита) или элементами и множествами для решения задач дискретной математики.

Теоретико-множественные операции необходимо заменять изоморфными логическими командами (and, or, not, xor) для последующего создания новой системы параллельного кубитного программирования для решения логических и оптимизационных задач.

Другое решение организации вычислительных процессов связано с топологическим представлением кубита, где элементами выступают геометрические фигуры.

Нечисленные задачи, ориентированные на предлагаемый процессор: минимизация форм (булевых функций) описания сложных систем; поиск путей в графе; тестирование и диагностика цифровых систем; комбинаторные исследования процессов и явлений; интеллектуальный поиск данных, распознавание образов и принятие решений; дискретизация фаззи моделей и методов в задачах создания интеллекта.

III. Заключение

Реализация квантового процессора на основе структуры Хассе позволила в n раз уменьшить аппаратурные затраты, по сравнению с число параллельной реализацией, что в свою очередь, уменьшило быстродействие процессора также в n раз. Вывод – необходимо создавать новые структуры данных для снижения аппаратной стоимости квантовых вычислений, и/или более интеллектуальные алгоритмы решения задачи покрытия на диаграммах Хассе. Синтез модели функциональности, представленной в форме одного куба, можно использовать при создании PLD-ориентированных прототипов вычислительных устройств, когда знание куба функциональности приводит к тривиальной технологии имплементации ее в кристалл.

Научная новизна – впервые предложена модель данных и структура аппаратной реализации квантового компьютера, которая характеризуется использованием структуры Хассе, что дает возможность существенно (x100) повысить быстродействие решения практических задач дискретной оптимизации.

Практическая значимость – существенное повышение быстродействия при решении задач покрытия и других задач дискретной оптимизации за счет увеличения аппаратных затрат для параллельного выполнения векторных логических операций на Хассе-структуре квантового вычислительного устройства.

IV. Библиография

1. Beth T. Quantum computing: an introduction // Proceedings of the IEEE International Symposium on Circuits and Systems, ISCAS.– 2000.– Geneva.– Vol. 1.– P. 735 – 736.

2. Jonker P., Jie Han. On quantum and classical computing with arrays of superconducting persistent current qubits // Proceedings of Fifth IEEE International Workshop on Computer Architectures for Machine Perception.–2000.– P. 69 – 78.

3. Keyes R.W. Challenges for quantum computing with solid-state devices // Computer.– Jan. 2005.– Vol. 38, Issue 1.– P. 65 – 69.

4. Glassner A. Quantum computing. 3. // IEEE Computer Graphics and Applications.– Nov/Dec 2001.– Vol. 21, Issue 6.– P. 72 – 82.

5. Marinescu D.C. The Promise of Quantum Computing and Quantum Information Theory – Quantum Parallelism // Proceedings of the 19th IEEE International Parallel and Distributed Processing Symposium (IPDPS'05).– 2005.– P. 112-114.

6. Oskin M., Chong F.T., Chuang I.L. A practical architecture for reliable quantum computers // Computer.– Jan. 2002.– Vol. 35, No.1.– P.79-87.

7. A. Glassner. Quantum Computing, Part 1 // IEEE Computer Graphics and Applications.– July/August 2001.– P.84-92.

8. P. Aliferis, F. Brito, D. P. DiVincenzo, J. Preskill, M. Steffen, B. M. Terhal. Fault-tolerant computing with biased-noise superconducting qubits // New Journal of Physics.– January 30, 2009.– 19 p. http://iopscience.iop.org/1367-2630/11/1/013061 /pdf/1367-2630_11_1_013061.pdf

9. B. Huntting, D. Mertz. Introduction to Quantum Computing // http://www.ibm.com/developerworks/linux/ library/l-quant/index.html

10. D. P. DiVincenzo. The Physical Implementation of Quantum Computation // IBM T.J. Watson Research Center, Yorktown Heights, NY 10598 USA.– July 9, 2004.– Preprint available online http://arxiv.org/PS_cache/ quant-ph/pdf/0002/0002077v3.pdf

11. Инфраструктура мозгоподобных вычислительных процессов / М.Ф. Бондаренко, О.А. Гузь, В.И. Хаханов, Ю.П. Шабанов-Кушнаренко.– Харьков: Новое Слово.– 2010.– 160 с.

12. Mark Gregory Whitney. Practical Fault Tolerance for Quantum Circuits. PhD Dissertation in Computer Science. Berkeley: University of California. 2009. 206p.

13. Хаханов В. И., Литвинова Е. И., Чумаченко С. В., Гузь О.А. Логический ассоциативный вычислитель. Электронное моделирование. № 1. 2011. С. 73-90.

14. Hahanov V., Wajeb Gharibi, Litvinova E., Chumachenko S. Information analysis infrastructure for diagnosis. Information. An international interdisciplinary journal. 2011. Japan. Vol.14. No 7. P. 2419-2433.

15. Хаханов В.И. Проектирование и тестирование цифровых систем на кристаллах / В.И. Хаханов, Е.И. Литвинова, О.А. Гузь. – Харьков: ХНУРЭ, 2009.– 484с.

16. Stig Stenholm, Kalle-Antti Suominen. Quantum approach to informatics. Published by John Wiley & Sons, Inc., Hoboken, New Jersey. 2005. 238p.

17. Michael A. Nielsen & Isaac L. Chuang. Quantum Computation and Quantum Information. Published in the United States of America by Cambridge University Press, New York. 2010. 676p.