Перчёнок Олег Владимирович

КВАНТОВОЕ МОДЕЛИРОВАНИЕ АЛГОРИТМА ДОЙЧА В СРЕДЕ MATLAB/SIMULINK

Аннотация

Статья посвящена имитационному моделированию квантового алгоритма Дойча в среде MATLAB/Simulink.

Ключевые слова: квантовый алгоритм Дойча, MATLAB, Simulink.

ВВЕДЕНИЕ

Необходимость построения квантового компьютера и разработки схемы квантовых вычислений возникает по двум причинам. Первая причина – технологическая. Фактическое развитие технологии изготовления больших интегральных схем неизбежно приводит к тому, что для записи бита классической информации требуются все более и более микроскопические объекты, по существу отдельные атомы и молекулы. Поведение этих объектов не укладывается в рамки классического описания, поэтому для дальнейшего развития таких технологий необходимо решать проблему организации вычислений на квантовых объектах.

Вторая причина, которая стимулирует исследования в области квантовых компьютеров, – это проявляющиеся принципиальные ограничения, которые возникают при использовании классических компьютеров для исполнения многих практически важных алгоритмов. Например, для факторизации (разложения на простые множители) 250-значного числа на существующих компьютерах потребуется примерно миллион лет! На квантовом компьютере разложение даже 1000-значного числа завершится всего за миллион шагов. Ввиду того, что данный алгоритм широко используется в криптографии, нельзя недооценивать возможность возникновения квантового компьютера. Кроме этого, принципиальным достоинством квантовых вычислений является возможность осуществления параллелизма вычислений, принципиально недоступного в классических устройствах. Пример данного свойства – возможность вычисления глобального свойства функции с помощью алгоритма Дойча (подробнее см. п. 1).

В Санкт-Петербургском Государственном Электротехническом Университете (СПбГЭТУ «ЛЭТИ») исследованиями в области квантовых компьютеров занимается кафедра Систем автоматического проектирования (САПР) факультета Компьютерных технологий и информатики (ФКТИ). Автор статьи под руководством старшего преподавателя кафедры САПР Матвеевой И.В. в ноябре–декабре 2009 года выполнил работу по моделированию нескольких квантовых объектов информации в среде MATLAB/Simulink. Среди них – квантовая цепь алгоритма Дойча, которому посвящена данная статья.

1. ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ [1; 2]

В классической теории состояние определяется заданием всех координат и скоростей составных частей системы в определенный момент времени. В квантовой теории *квантовое состояние* – это полный набор данных (физических величин), определяющих свойства системы. Какие именно данные его определяют, зависит от конкретной системы. Так как состояние – совокупный набор данных, то можно сказать, что состояние есть вектор.

В квантовой теории информации аналогом классического бита является *квантовый бит (кубит).* Классический бит обладает двумя *булевыми состояниями* 0 или 1, принимаемыми с вероятностями либо $P(0) = 1$, либо $P(1) = 1$. Кубит, в отличие от него, находится одновременно сразу в обоих своих базисных состояниях, то есть образует *когерентную суперпозицию базисных квантовых состояний*, которая в случае *чистого* (когерентного) квантового состояния описывается одним из двух эквивалентных способов, а именно:

а) *вектором состояния*, представляемым вектором-столбцом длиной 2:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle = \begin{pmatrix}\alpha\\ \beta\end{pmatrix},\ |\alpha|^2 + |\beta|^2 = 1, \quad (1)$$

где α, β – комплексные амплитуды, $|\alpha|^2$, $|\beta|^2$ – вероятности, с которыми кубит находится в базисных состояниях $|0\rangle = \begin{pmatrix}1\\0\end{pmatrix}$ и $|1\rangle = \begin{pmatrix}0\\1\end{pmatrix}$.

б) *матрицей плотности*:

$$\mathbf{D} = \begin{pmatrix}|\alpha|^2 & \alpha\beta^*\\ \alpha^*\beta & |\beta|^2\end{pmatrix}. \quad (2)$$

Любая логическая операция с кубитами называется *квантовым вентилем* (*преобразователем*), или (в англоязычной литературе) *гейтом* (англ. gate – ворота). По числу кубитов преобразователи делятся на одно- и многокубитные. Преобразователь переводит одно состояние кубита (а в многокубитном случае – квантового регистра) в другое.

Для демонстрации действия квантового преобразователя на кубиты используют матричную запись оператора.

Для того чтобы получить вектор состояния результирующего кубита $|\psi_{out}\rangle$, необходимо умножить вектор состояния исходного кубита $|\psi_{out}\rangle$ на матрицу преобразования **M**:

$$|\psi_{in}\rangle = |\psi_{out}\rangle \cdot \mathbf{M}. \quad (3)$$

Для того чтобы получить матрицу плотности результирующего регистра $\mathbf{D_{out}}$ из матрицы плотности исходного регистра $\mathbf{D_{in}}$ с помощью матрицы преобразования **M**, необходимо воспользоваться следующей формулой:

$$\mathbf{D_{out}} = \mathbf{M} \cdot \mathbf{D_{in}} \cdot \mathbf{M}^T. \quad (4)$$

Рассмотрим однокубитные гейты: *Адамара* и NOT. Приведем их матрицы (**H** и **NOT** соответственно):

$$\mathbf{H} = \frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix};\ \mathbf{NOT} = \begin{pmatrix}0 & 1\\ 1 & 0\end{pmatrix}.$$

Набор N кубитов составляет *квантовый регистр*. Принципиальным моментом теории является то, что этот регистр в общем случае также подчиняется принципу суперпозиции и находится одновременно во всех своих базовых классических состояниях, число которых $Q = 2^N$.

Общее состояние такой системы кубитов описывается тензорным произведением входящих в нее кубитов. Для матриц произведение Кронекера (тензорное) выполняется по правилу:

$$\mathbf{A} \otimes \mathbf{B} = \begin{pmatrix}a & b\\ c & d\end{pmatrix} \otimes \mathbf{B} = \begin{pmatrix}a\mathbf{B} & b\mathbf{B}\\ c\mathbf{B} & d\mathbf{B}\end{pmatrix}. \quad (5)$$

Одним из базовых двухкубитных преобразователей является преобразователь контролируемое НЕ (CNOT). При своем воздействии на двухкубитный регистр этот оператор оставляет первый кубит (Control) без изменения, а второй (Target) инвертирует, если Control равен $|1\rangle$, иначе оставляет без изменения. На рис. 1 помещено схематическое изображение CNOT. Приведем матрицу CNOT:

$$\mathbf{CNOT} = \begin{pmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 1\\ 0 & 0 & 1 & 0\end{pmatrix}.$$

Далее перейдем к рассмотрению задачи Дойча и алгоритма, обеспечивающего ее решение [2].

Пусть имеются четыре бинарные функции $f_i(x)$ от двоичной переменной $x = 0,1$. При этом функции f_1 и f_2 постоянны и принимают значения $f_1(x) = 0$, $f_2(x) = 1$, функции f_3 и f_4 принимают значения $f_3(x) = x$, $f_4(x) = \neg x$. Говорят, что функции

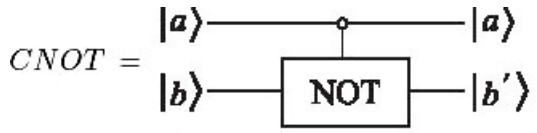

Рис. 1. Оператор CNOT

f_3 и f_4 сбалансированы. В задаче Дойча требуется определить, к какой группе (постоянные или сбалансированные) относится функция $f_i(x)$.

Классическое решение такой задачи предполагает проведение как минимум двух операций – вычисление $f_i(0)$ и $f_i(1)$. Квантовый алгоритм позволяет решить задачу за одну операцию – *квантовую цепь* (цепь из квантовых гейтов) алгоритма Дойча (рис. 2).

На входе цепи находится двухкубитный регистр, состояние которого постоянно:

$$|\psi_0\rangle = |0\rangle \otimes |1\rangle = |01\rangle .$$

Сначала на каждый из кубитов входного регистра действуем преобразованием Адамара **H**. Далее к получившемуся регистру применяем оператор **U**, который представляется четырьмя матрицами размерности 4×4, определяющими четыре возможные функции $f_i(x)$:

$$\mathbf{U}_{\mathbf{f}_1} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} = \hat{\mathbf{I}} \otimes \hat{\mathbf{I}};$$

$$\mathbf{U}_{\mathbf{f}_2} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} = \hat{\mathbf{I}} \otimes \mathbf{NOT};$$

$$\mathbf{U}_{\mathbf{f}_3} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} = \mathbf{CNOT};$$

$$\mathbf{U}_{\mathbf{f}_4} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} = \mathbf{CNOT}(\hat{\mathbf{I}} \otimes \mathbf{NOT}),$$

где $\hat{\mathbf{I}} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ – единичная матрица.

И, наконец, воздействуя на первый из результирующих кубитов оператором Адамара, получим выходной регистр, состояние которого зависит от функции $f(x)$:

$$|\psi_3\rangle = \pm| f(0) \oplus f(1)\rangle \otimes \frac{|0\rangle - |1\rangle}{\sqrt{2}} .$$

Таким образом, измеряя состояние первого выходного кубита, можно получить значение $f(0) \oplus f(1)$, то есть квантовая цепь дает возможность определить глобальное свойство функции $f(x)$, используя только одно вычисление $f(x)$. Это явление называется *квантовым параллелизмом*.

Если состояние первого кубита на выходе цепи $|0\rangle$, то $f_{1,2}(x)$ – постоянные функции. Если состояние первого кубита на выходе цепи $|1\rangle$, то $f_{3,4}(x)$ – сбалансированные функции. Существенно, что при этом не вычисляются значения самих функций. Квантовый компьютер выделяет «глобальную» информацию из суперпозиции состояний.

2. ОПИСАНИЕ СРЕДЫ МОДЕЛИРОВАНИЯ [3]

MATLAB (сокращение от англ. *«Matrix Laboratory»*) – термин, относящийся к пакету прикладных программ фирмы Mathworks Inc. для решения задач технических вычислений и имитационного моделирования, а также к используемому в этом пакете языку программирования.

Язык MATLAB является высокоуровневым интерпретируемым языком программирования, включающим основанные на матрицах структуры данных, широкий спектр функций, интегрированную среду разработ-

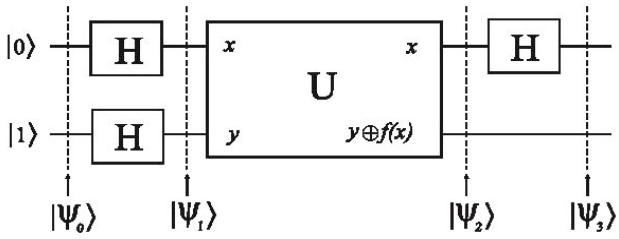

Рис. 2. Квантовая цепь алгоритма Дойча

ки, объектно-ориентированные возможности и интерфейсы к программам, написанным на других языках программирования.

Программы, написанные на MATLAB, бывают двух типов – функции и сценарии (англ. *scripts*). Функции имеют входные и выходные аргументы, а также собственное рабочее пространство для хранения промежуточных результатов вычислений и переменных. Сценарии же используют общее рабочее пространство. Как сценарии, так и функции не компилируются в машинный код и сохраняются в виде текстовых файлов. Основной особенностью языка MATLAB является его широкие возможности по работе с матрицами, которые создатели языка выразили в лозунге «думай векторно».

Simulink – это компонент среды MATLAB, предназначенный для структурно-функционального имитационного моделирования и анализа динамических систем. Он состоит из инструмента составления блочных диаграмм и изменяемого набора библиотек блоков (элементов). Simulink позволяет провести интеграцию построенных диаграмм с остальной средой MATLAB. Возможен вызов на выполнение моделей Simulink из сценариев и функций MATLAB и вызов функций MATLAB из блоков Simulink. Компонент широко используется в области теории управления и цифровой обработки сигналов для моделирования и проектирования.

3. ОПИСАНИЕ МОДЕЛИ

Модель квантовой цепи алгоритма Дойча была построена в Simulink. Изобразим ее на рис. 3.

На вход цепи подаются вектора состояний двух кубитов Q1 и Q2 и номер функции (Number of function) i. На каждый из кубитов действуем оператором Адамара (функция H1, см. п. 4.1). Далее к преобразованным кубитам применяем оператор $\mathbf{U}_{\mathbf{f}_i}$. Данное преобразование реализуется с помощью функции apply_Uf (см. п. 4.2). И наконец, к первому из результирующих кубитов применяем оператор Адамара (функция H_on_1, см. п. 4.3). На выходе получаем вектора состояний двух результирующих кубитов Q1' и Q2'.

Модель может функционировать в двух режимах.

Режим «калькулятора» (РК) – режим демонстрации модели, при котором входные величины вводятся непосредственно во входные блоки, а запуск модели происходит в окне модели Simulink. Выходные значения изображаются в блоках Display, напоминающих дисплей калькулятора.

Режим временных диаграмм (РВД) – режим демонстрации модели, при котором входные величины вводятся во входные блоки через сценарий визуализации, который изображает временные диаграммы. Непосредственно в блоки ставятся имена переменных. Запуск модели осуществляется в

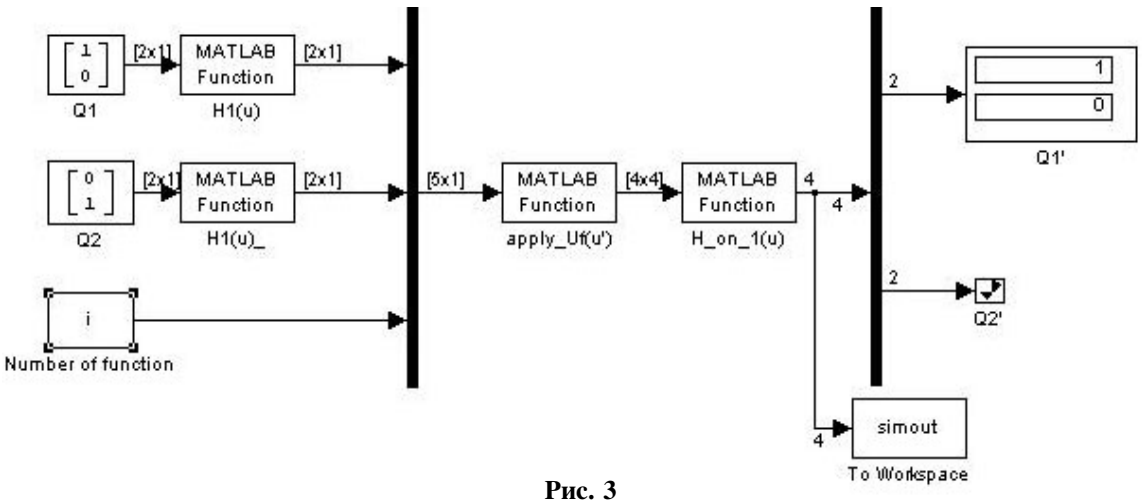

Рис. 3

среде MATLAB с помощью сценария визуализации.

Для ввода исходных кубитов используется элемент Constant, в котором указываются непосредственно векторы состояния кубитов. Для ввода номера функции используется элемент Constant, в котором указывается непосредственно число (при функционировании модели в РК) или переменная i (при функционировании модели в РВД).

Для вызова функций H1, apply_Uf, H_on_1 используются блоки MATLAB Function.

Для вывода выходных кубитов Q1' и Q2' используются блоки Display (обеспечивают РК) и блок To Workspace (обеспечивает вывод объединенного вектора из векторов состояния Q1', Q2' в переменную simout для демонстрации модели в РВД). Блок Display, выводящий кубит Q2' свернут, так как значение вектора состояния этого кубита нас не интересует.

4. ОПИСАНИЕ ФУНКЦИЙ MATLAB, ИСПОЛЬЗУЕМЫХ В МОДЕЛИ

4.1. ФУНКЦИЯ H1

Для осуществления преобразования Адамара разработана функция H1.

Формат вызова: Q1=H1(Q).

Параметры: Q – вектор состояния исходного кубита.

Возвращаемое значение: Q1 – вектор состояния выходного кубита.

Приведем листинг функции H1 (см. листинг 1).

Описание алгоритма.

Осуществляется вычисление вектора состояния выходного кубита на основе вектора состояния входного кубита и матрицы Адамара по формуле (3).

В функции используется глобальная переменная (фактически константа) Hadamard. Она представляет собой матрицу Адамара и, как и все последующие используемые константы, задается в специально разработанной функции define_globals.

4.2. ФУНКЦИЯ APPLY_UF

Для осуществления преобразования разработана функция apply_Uf.

Формат вызова: newstate = apply_Uf(QQ)

Параметры: QQ – вектор, представляющий собой последовательное объединение векторов состояний исходных кубитов и номера функции.

Возвращаемое значение: newstate – матрица плотности выходного регистра.

Приведем листинг функции apply_Uf (см. листинг 2).

Описание алгоритма.

Формируем трехмерный массив, другими словами, массив матриц операторов.

Из частей исходного объединенного вектора QQ с помощью специально разработанной функции density, действующей по формуле (2), получаем матрицы плотности исходных кубитов, выделяем номер функции i. Далее посредством встроенной функции MATLAB kron, реализующей тензорное произведение матриц – формула (5) – формируем матрицу плотности входного регистра. На полученную матрицу действуем оператором по формуле (4), извлекая нужную матрицу из трехмерного массива.

4.3. ФУНКЦИЯ H_ON_1

Для воздействия оператором Адамара на первый кубит в двухкубитном регистре разработана функция H_on_1.

Листинг 1

```
%  h1.m  однокубитное преобразование Адамара
function Q1 = H1(Q)
    global Hadamard;
    Q1=Hadamard*Q;
```

Листинг 2

```
%  apply_Uf.m Применение операторов Uf1-Uf4 в алгоритме Дойча
function newstate = apply_Uf(QQ)
    global I2x2; global NotGate; global CNotGate;
    % определение операторов постоянных функций
    Uf(:,:,1)=eye(4);
    Uf(:,:,2)=kron(I2x2,NotGate);
    % определение операторов сбалансированных функций
    Uf(:,:,3)=CNotGate;
    Uf(:,:,4)=CNotGate*Uf(:,:,2);
    pq1=density(QQ(1:2)); % получение матриц плотностей исходных кубитов
    pq2=density(QQ(3:4)); % по векторам состояния
    i=QQ(5); % получение заданного номера функции
    state = kron(pq1,pq2); % получение состояния исходного регистра
    % применение оператора заданной функции
    newstate = Uf(:,:,i) * state * Uf(:,:,i)';
```

Формат вызова: newQQ=H_on_1(state1)

Параметры: state1 – матрица плотности входного регистра

Возвращаемое значение: newQQ – вектор, представляющий собой последовательное объединение векторов состояний выходных кубитов.

Приведем листинг функции H_on_1 (см. листинг 3).

Описание алгоритма.

Формируем матрицу M оператора с помощью функции kron из матрицы Адамара и единичной матрицы. На матрицу плотности исходного регистра действуем оператором M по формуле (4). После этого из результирующей матрицы выделяем плотности выходных кубитов с помощью функции den_from_state, из которых, в свою очередь, восстанавливаем вектора состояний посредством функции restore_qubit. И, наконец, формируем выходной объединенный вектор из векторов состояний кубитов результирующего регистра.

Функции den_from_state и restore_qubit разработаны автором статьи.

5. ДЕМОНСТРАЦИЯ РАБОТЫ МОДЕЛИ

Приведем извлечение из протокола тестирования модели в РК:

Проделана следующая последовательность действий:

1. Вызов функции define_globals из командной строки MATLAB.

2. Открытие файла модели Simulink «Deutsch.mdl».

3. Перебор всех возможных значений входных номера функции в модели Simulink:

Листинг 3

```
% H_on_1.m  Воздействие преобразователя Адамара на первый кубит
% в двухкубитном состоянии
function newQQ=H_on_1(state1)
    global Hadamard; global I2x2;
    M=kron(Hadamard,I2x2); % подготовка матрицы оператора
    state = M * state1 * M'; % применение оператора
    pq1=den_from_state(1,state); % получение матриц плотностей кубитов из
    pq2=den_from_state(2,state); % состояния результирующего регистра
    q1=restore_qubit(pq1); % получение векторов состояния из
    q2=restore_qubit(pq2); % матриц плотности
    newQQ=[q1 q2]';
```

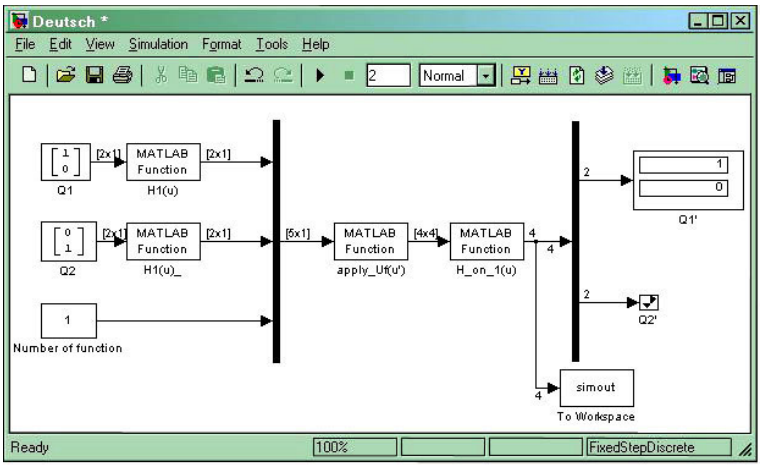

Рис. 4. Вид окна модели
при тестировании в режиме «калькулятора»

1,2,3,4. Для каждого значения осуществлялся его ввод в блок Constant «Number of function» модели Simulink и производился запуск. После окончания работы модели на верхнем блоке Display отображались результаты, совпадающие с теоретическими. На рис. 4 изобразим вид окна модели после запуска со входным значением i = 1.

Перейдем к описанию демонстрации модели в РВД. Для данной демонстрации был разработан сценарий визуализации, выводящий временные диаграммы векторов состояний входных кубитов, номера функции и вектора состояния первого выходного кубита. Приведем извлечение из протокола тестирования модели в РВД:

Проделана следующая последовательность действий:

1. Открытие файла сценария визуализации «DeutschSim.m» в среде MATLAB.

2. Запуск сценария. В процессе его отработки сформировалось окно «Figure 1» («Изображение 1»). Содержимое окна представляет собой временные диаграммы (см. рис. 5). По диаграммам установлено, что результаты моделирования совпадают с теоретическими.

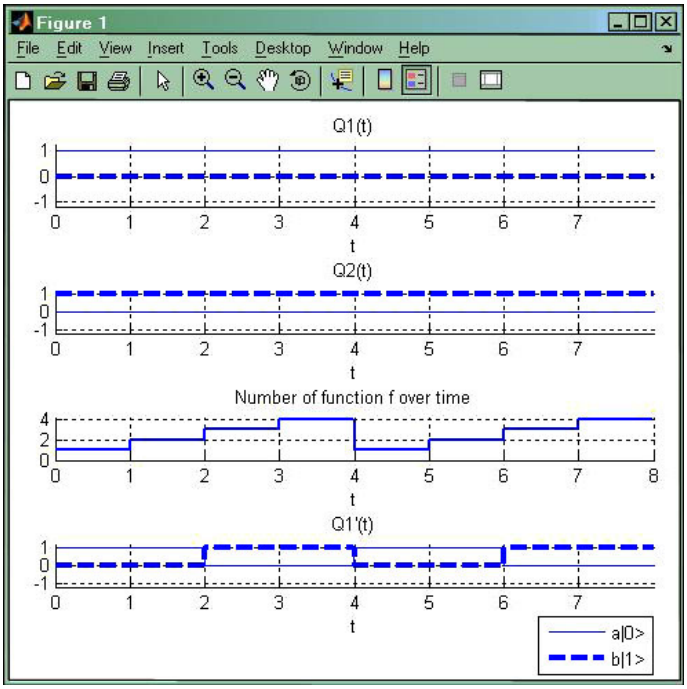

Рис. 5. Вид окна изображения при тестировании
в режиме вектроных диаграмм

ЗАКЛЮЧЕНИЕ

У читателя может возникнуть вопрос: «Почему для моделирования использовалась именно среда MATLAB/Simulink?» Действительно, существуют и другие средства, которые используются для моделирования квантовых объектов (так называемые квантовые симуляторы). Например [4]:

- Библиотеки функций для языков программирования (C/C++, Java, LISP и др.)
- Симуляторы, созданные в математических пакетах (Maple, Mathematica и др.)
- Online-симуляторы (Quantum eXpress, GQC и др.)

Однако перечисленные выше средства не столь известны среди ученых различного профиля, как

MATLAB. Этот инструмент используют более миллиона инженерных и научных работников, он функционирует на большинстве современных операционных систем, включая GNU/Linux, Mac OS, Solaris и Microsoft Windows [3].

Существуют также библиотеки функций и сценариев MATLAB, моделирующие работу квантового компьютера [4] (например, CS 596 Quantum Computing, QLib). Но в этих решениях не используется Simulink, из-за этого они менее наглядны, чем описанная работа.

Привлечение специалистов из разных областей к исследованиям, связанным с квантовым компьютером, ускорило бы приближение к конечной цели – физической реализации квантового компьютера. Данная статья вносит некоторый вклад в такую «PR-кампанию». Кроме этого, MATLAB изучают во многих технических вузах. Введение курса лабораторных работ по квантовому моделированию в среде MATLAB вовлекло бы заинтересовавшихся студентов в эту область информатики будущего, которое постепенно становится настоящим.

Литература

1. *Герасимов И.В., Калмычков В.А., Лозовой Л.Н.* Комплементарное моделирование в средах САПР: виртуализация квантовых объектов информации. СПб.: Изд-во СПбГЭТУ «ЛЭТИ», 2007.

2. *Deutsch D.* Quantum Theory the Church-Turing Principle and the Universal Quantum Computer. Proc. Roy. Soc. London, 1985, V A400, # 1818, p. 57 / Перевод с англ. под ред. *В.А. Садовничего*: Сборник «Квантовый компьютер и квантовые вычисления». Ижевск. Ред. журн. «Регулярная и хаотическая динамика», 1999.

3. Википедия – свободная энциклопедия: http://www.wikipedia.org

4. Quantiki – энциклопедия по квантовому компьютеру: http://www.quantiki.org/wiki/index.php/List_of_QC_simulators

5. *И.В. Герасимов, В.А. Калмычков, И.В. Матвеева, Л.А. Чугунов.* Представление данных, исследования и визуализация в среде «Matlab». Практикум по дисциплине: «Информатика». СПб.: Изд-во СПбГЭТУ «ЛЭТИ», 2006.

6. *Потемкин В.Г.* MATLAB 6: среда проектирования инженерных приложений. М.: Диалог-МИФИ, 2003.

Abstract

The article is devoted to quantum Deutsch's algorithm simulation in the MATLAB/Simulink environment.

Keywords: quantum Deutsch's algorithm, MATLAB, Simulink.

Перчёнок Олег Владимирович,
магистрант 2 курса магистратуры СПбГЭТУ «ЛЭТИ», факультет Компьютерных технологий и информатики (ФКТИ), кафедра Автоматики и процессов управления (АПУ)

olegperch@mail.ru