Управление двухкубитной системой

Модель

Рассмотрим систему из двух кубитов: Q_1 с частотой $\omega_{01}^{(1)}$ и ангармонизмом $\mu^{(1)}$ и Q_2 с частотой $\omega_{01}^{(2)}$ и ангармонизмом $\mu^{(2)}$ ($\mu^{(n)} = \omega_{01}^{(n)} - \omega_{12}^{(n)}$). Кубиты управляются посредством внешних электромагнитных сигналов, причём каждый кубит подсоединён к своей системе управления – генератору SFQ импульсов – через соединительную ёмкость C_c. Кубиты соединены между собой посредством связи, которую для простоты будем характеризовать одним параметром – силой связи g (примечание: это приближение хорошо работает не только для емкостных связей, но и для соединений кубитов через высокодобротный резонатор). Поскольку нас интересует утечка на вышележащие уровни, мы будем работать в трёхуровневом приближении (состояние одиночного кубита выражается матрицей 1x3, двух кубитов – 1x9, оператор однокубитной операции – матрицей 3x3, двухкубитной – 9x9).

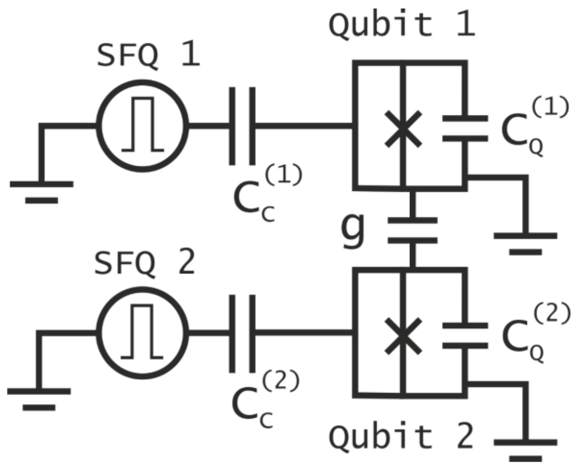


Рис. 1. Схема SFQ-управления двухкубитной системой

Гамильтонианы кубитов ($\mathbb{I}$ – единичная матрица размера 3x3):

$$H_1 = \hbar\omega_{01}^{(1)} a^\dagger a - \frac{\hbar\mu^{(1)}}{2} a^\dagger a\left(a^\dagger a - \mathbb{I}\right)$$

$$H_2 = \hbar\omega_{01}^{(2)} b^\dagger b - \frac{\hbar\mu^{(2)}}{2} b^\dagger b\left(b^\dagger b - \mathbb{I}\right)$$

Здесь используются операторы вторичного квантования: $a^\dagger$ и $b^\dagger$ – операторы рождения, a и b – операторы уничтожения.

$$a^\dagger = b^\dagger = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & \sqrt{2} & 0 \end{pmatrix}, a = b = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & \sqrt{2} \\ 0 & 0 & 0 \end{pmatrix}$$

Гамильтониан кубита с номером n:

$$H_n = \begin{pmatrix} 0 & 0 & 0 \\ 0 & \hbar\omega_{01}^{(n)} & 0 \\ 0 & 0 & 2\hbar\omega_{01}^{(n)} - \hbar\mu^{(n)} \end{pmatrix}$$

Гамильтониан общей двухкубитной системы (H_{2q}) складывается из двух частей: собственных гамильтонианов кубитов, приведённых к нужной размерности, и слагаемого, ответственного за межкубитное взаимодействие через связь с силой g (H_{int}):

$$H_{2q} = H_1 \otimes \mathbb{I} + \mathbb{I} \otimes H_2 + H_{int}$$

$$H_{int} = \hbar g\left(a^\dagger + a\right) \otimes \left(b^\dagger + b\right)$$

В случае управления посредством электромагнитного поля добавляются слагаемые, соответствующие управляющему полю. В нашем случае, это импульсы, подаваемые с двух генераторов:

$$H_{full} = H_{2q} + V_1 \otimes \mathbb{I} + \mathbb{I} \otimes V_2$$

Для случая SFQ-управления, воздействие одиночного импульса на кубит с номером *n* представляется через гамильтониан (знак зависит от полярности импульса):

$$V_n = \pm\, iC_c^{(n)} V_0 \sqrt{\frac{\hbar\omega_{01}^{(n)}}{2C_Q^{(n)}}} \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & -\sqrt{2} \\ 0 & \sqrt{2} & 0 \end{pmatrix},$$

где $V_0 = \Phi_0/w$, Φ_0 – квант магнитного потока, w – ширина импульса.

Все входные параметры системы можно условно разделить на два типа: заданные и регулируемые. К заданным относятся параметры, экспериментально полученные посредством измерений изготовленной системы, значение которых невозможно изменить. К регулируемым относятся параметры, значение которых можно изменять для управления двухкубитной системой.

Заданные параметры		**Регулируемые параметры**
Параметр	**Типичное значение**	**Параметр**
Основная частота кубита, ω_{01}	3-7 ГГц	Частота генератора, ω_g
Ангармонизм кубита, μ	0.2-0.5 ГГц	Длительность импульса, τ
Собственная емкость кубита, C_Q	1-10 пФ	Смещение между генераторами, φ
Соединительная емкость, C_C	0.1-10 фФ	Вид сигнала в виде последовательности
Сила связи, g	0.01-0.2 ГГц	

Код для управления регулярными/нерегулярными последовательностями импульсов

Для моделирования управления двухкубитной системой был написан код на языке MATLAB. Он состоит из нескольких файлов:

- main.m – основной исполняемый скрипт с указанием входным параметров системы
- SimulateRegular.m – вычислительная часть для управления регулярными последовательностями импульсов
- SimulateRegularDraw.m – то же самое, но населённости считаются на каждом шаге сетки для визуализации динамики
- SimulateIrregular.m – вычислительная часть для управления нерегулярными последовательностями импульсов
- SimulateIrregularDraw.m – то же самое, но населённости считаются на каждом шаге сетки для визуализации динамики
- UMatrix.m – функция для преобразования гамильтониана в оператор эволюции

Для регулярного управления в main задаются следующие *входные параметры*:

- *N* – количество учитываемых уровней на одном кубите (пока работает для 2 и 3)
- *tstep* – шаг разбиения сетки
- *w1* – собственная частота кубита 1
- *w2* – собственная частота кубита 2
- *mu1* – ангармонизм кубита 1
- *mu2* – ангармонизм кубита 2
- *g* – сила межкубитной связи
- *Cq1* – собственная ёмкость кубита 1
- *Cq2* – собственная ёмкость кубита 2
- *Cq1* – емкость между кубитом 1 и генератором 1
- *Cq2* – емкость между кубитом 2 и генератором 2

- *wg1* – частота генератора 1
- *wg2* – частота генератора 2
- *tau* – ширина импульса
- *phi* – задержка между двумя генераторами (в шагах сетки, задержка на генераторе 1)
- *waitq1* – ожидание после применения последовательности 1 (в шагах сетки)
- *waitq2* – ожидание после применения последовательности 2 (в шагах сетки)
- *init* – начальное состояние двухкубитной системы, например, '01'
- *operation* – идеальная операция, относительно которой считается фиделити
    - *Для регулярных последовательностей:*
- *N1* – количество импульсов, подаваемых с генератора 1
- *N2* – количество импульсов, подаваемых с генератора 2
- *bip1* – тип последовательности 1 (0 – униполярный, 1 – биполярный)
- *bip2* – тип последовательности 2 (0 – униполярный, 1 – биполярный)

Для биполярных импульсов количество учитывается для импульсов обеих полярностей.

- *Для нерегулярных последовательностей:*
- str1 – последовательность, подаваемая на кубит 1
- str2 – последовательность, подаваемая на кубит 2

Алгоритм вычисления:

1. Нахождение ВФ, соответствующих базисным состояниям двухкубитной системы
2. Вычисление операторов эволюции, соответствующих разным случаям подаваемых импульсов на двухкубитную систему, при помощи OperatorGrid (см. ниже)
3. Составление общего оператора эволюции всей последовательности
4. Применение оператора эволюции к начальному состоянию, вычисление населённостей уровней системы после применения операции и fidelity (фиделити пока вычисляется неправильно)

Для выполнения пункта 2 используется специальная функция OperatorGrid. Её суть в том, чтобы понять, какой оператор необходимо применить к двухкубитной системе на выбранном шаге сетки в зависимости от параметров регулярной последовательности. OperatorGrid составляет две временные сетки для каждой последовательности, а затем объединяет их в одну и на выходе выдаёт текстовую строку с номерами нужных операторов эволюции. Оператор эволюции имеет вид U_{nk}, где n – импульс в момент в момент времени t на кубите 1, k – импульс в момент в момент времени t на кубите 2. n и k могут иметь значение 0, 1 или -1 в зависимости от наличия и типа импульса в момент времени t.

Полученная при помощи OperatorGrid строка расшифровывается следующим образом:

Вывод OperatorGrid	0	1	2	3	4	5	6	7	8
Соответствующий оператор	U_{00}	U_{01}	U_{10}	U_{11}	U_{-10}	U_{0-1}	U_{-11}	U_{1-1}	U_{-1-1}

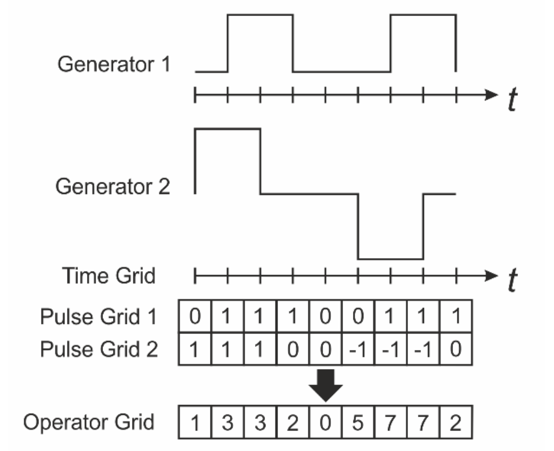


Рис. 2 Наглядный пример работы OperatorGrid

Для примера возьмём большую сетку. Первый генератор подаёт униполярные импульсы шириной 2 grs (grs = шаг сетки), расстояние между импульсами – 3 grs, задержка перед последовательностью – 1grs. Второй – биполярные импульсы такой же ширины и с такой же частотой, но без задержки. На pulse grid показывается какой импульс стоит в узлах сетки, на operator grid – какое значение примет функция OperatorGrid в узлах общей сетки.

Пример вычисления регулярных последовательностей:

Зададим следующие начальные параметры (val = $2\pi * 10^9$):

N	tstep	w1	w2	mu1	mu2	g	Cq1	Cq2	Cc1	Cc2
3	5e-14	5*val	5.2*val	0.25*val	0.4*val	0.02*val	1e-12	1e-12	4e-16	4e-16
N1	**N2**	**wg1**	**wg2**	**tau**	**phi**	**waitq1**	**waitq2**	**bip1**	**bip2**	**init**
80	0	5*val	5.2*val	4e-12	0	0	0	0	0	‘00’

Результат применения SimulateRegular:

00: 0.48953

10: 0.50945

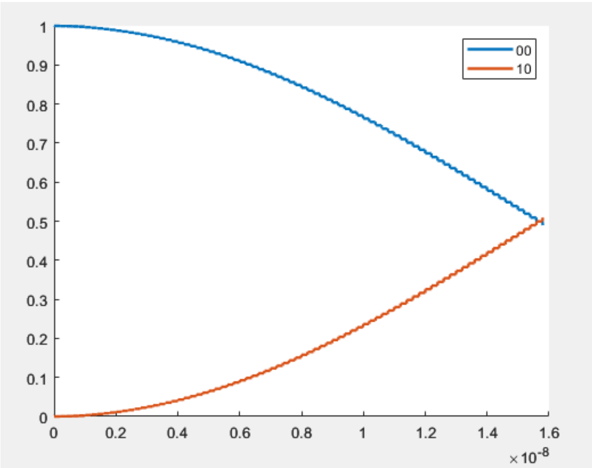
01: 9.2458e-06

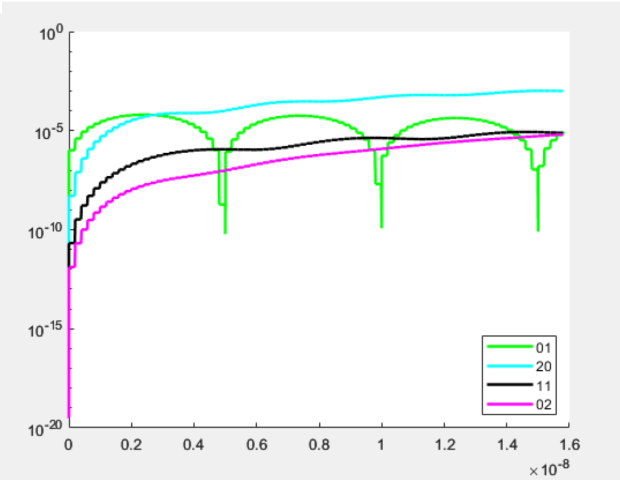
20: 0.00099703

02: 6.5567e-06

11: 7.4048e-06

Результат применения SimulateRegularDraw:





Теперь то же самое, но поменяем bip1 = 1

Результат применения SimulateRegular:

00: 0.499

10: 0.49704

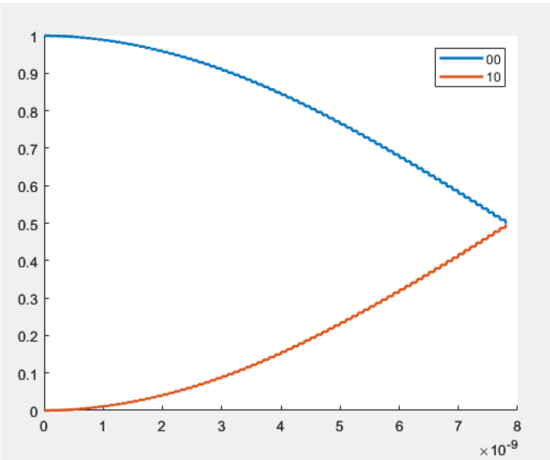
01: 0.00013764

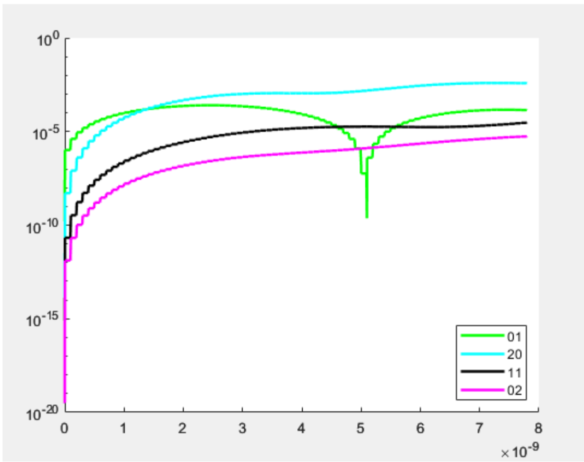
20: 0.0037816

02: 5.6523e-06

11: 3.0661e-05

Результат применения SimulateRegularDraw:





Пример вычисления нерегулярных последовательностей:

Зададим следующие начальные параметры (val = $2\pi * 10^9$):

N	tstep	w1	w2	mu1	mu2	g	Cq1	Cq2	Cc1	Cc2
3	5e-14	5*val	5.2*val	0.25*val	0.4*val	0.02*val	1e-12	1e-12	4.9e-16	4e-16
		wg1	**wg2**	**tau**	**phi**	**waitq1**	**waitq2**	**init**		
		25*val	25*val	4e-12	0	0	0	‘00’		
str1										
11-1-1-1110-1011-1-1-111-1-1-111-1-1110-1-1110-1-1110-1-111-10-1111-1-1110-1-1111-1-1111-1-1111 -1-1011-10111-1-1011-1-1-111-1-1-111-10110-1-111-1-1-101-1-1-111-1-1-1110-1-11										

Результат применения SimulateIrregular:

00: 0.48959

10: 0.51

01: 6.1892e-05

20: 4.5642e-06

02: 4.7072e-06

11: 0.00031275

Результат применения SimulateIrregularDraw: