Quantum Fuzzy Modeling System: SW&HW Support of Quantum Computational Intelligence and Intelligent Control

I.S. Ulyanov* and *S.V. Ulyanov
MCG, "*Quantum*" Ltd, Co. Moscow ulyanovsv@mail.ru

Abstract Design of flexible structure for SW/HW support of Quantum Modeling System (QMS) is discussed. Functional properties of QMS are described from viewpoint of quantum algorithm (QA) theory. Structure of QMS includes examples of decision-making and search QAs, typical quantum operators (superposition, entanglement, quantum oracles, interference etc.), and design process of quantum control algorithms. Application of QMS in design of intelligent control system is discussed.

Key words: *Quantum modeling, quantum control algorithm, intelligent control*

Introduction

Computational intelligence is one of an effective toolkit for fuzzy modeling system in design technology of robust intelligent control systems. We have developed a new quantum fuzzy modeling system (QFMS) based on a new computational intelligence paradigm as quantum computing technology for design of self-organization robust KB in unpredicted control situations [1]. Computation, based on the laws of classical physics, leads to different constraints on information processing than computation based on quantum mechanics. Quantum computers hold promise for solving many intractable problems. But, unfortunately, there currently exist no algorithms for "programming" a quantum computer. Calculation in a quantum computer (like calculation in a conventional computer) can be described as a marriage of quantum HW (the physical embodiment of the computing machine itself, such as quantum gates and the like), and quantum SW (the computing algorithm implemented by the HW to perform the calculation). To date, quantum SW algorithms, such as Shor's algorithm, used to solve problems on a quantum computer have been developed on an *ad hoc* basis without any real structure or programming methodology.

Important computer-scientific challenges for quantum information science are to discover efficient QAs for interesting problems and to understand the fundamental capabilities and limitations of quantum computation in comparison to those of classical computation. The bulk of this report is concerned with the problem of discovering new QAs. Perhaps the most important open problem in the theory of quantum information processing is to understand the nature of quantum mechanical speed-up for the solution of computational problems:

What problems can be solved more rapidly using quantum computers than is possible with classical computers, and what ones cannot?

To take full advantage of the power of quantum computers, we should try to find new problems that are amenable to quantum speed-up. More importantly, we should try to broaden the range of available algorithmic techniques for quantum computers, which is presently quite limited. The first examples of problems that can be solved faster with a quantum computer than with a classical computer were *oracular*, or *black-box*, problems. In standard computational problems, the input is simply a string of data such as an integer or the description of a graph. In contrast, in the black-box model, the computer is given access to a black box, or oracle that can be queried to acquire information about the problem. The goal is to find the solution to the problem using as few queries to the oracle as possible. This model has the advantage that proving lower bounds is tractable, which allows one to demonstrate provable speed-up over classical algorithms, or to show that a given QA is the best possible.

Related works [2, 3]. *Deutsch*'s pioneering example of quantum speed-up was an oracular problem that can be solved on a quantum computer using one query, but that requires two queries on a classical computer. *Deutsch* and *Jozsa* generalized this problem to one that can be solved exactly on a quantum computer in polynomial time, but for which an exact solution on a

classical computer requires exponential time. However, the *Deutsch-Jozsa* problem can be solved with high probability in polynomial time using a probabilistic classical algorithm. *Bernstein* and *Vazirani* gave the first example of a superpolynomial separation between probabilistic classical and quantum computation, and Simon gave another example in which the separation is exponential. This sequence of examples of rather artificial oracular problems led to Shor's aforementioned discovery of efficient QAs for the factoring and discrete log problems, two non-oracular computational problems with practical applications, for which no polynomial-time classical algorithm is known. *Shor*'s algorithm, like its predecessors, is based on the ability to efficiently implement a quantum Fourier transforms. More recently, numerous generalizations and variations of *Shor*'s algorithm have been discovered for solving both oracular and non-oracular problems with superpolynomial speed-up. All of these algorithms are fundamentally based on quantum Fourier transforms (QFT). A second tool used in quantum algorithms comes from *Grover*'s algorithm for unstructured search. In the unstructured search problem, one is given black box access to a list of N items, and must identify a particular marked item. Classically, this problem clearly requires $O(N)$ queries, but *Grover* showed that it could be solved on a quantum computer using only $O\left(\sqrt{N}\right)$ queries (which had previously been shown to be the best possible result by *Bennett, Bernstein, Brassard,* and *Vazirani*). This speed-up is more modest than the speed-up of *Shor*'s algorithm—it is only quadratic rather than superpolynomial—but the basic nature of unstructured search means that it can be applied to a wide variety of other problems. *Grover*'s algorithm was subsequently generalized to the concept of amplitude amplification, and many extensions and applications have been found. The *Shor* and *Grover* algorithms and their relatives will surely be useful if large-scale quantum computers can be built. But they also raise the question of how broadly useful quantum computers could be. It appears to be difficult to design QAs, so it would be useful to have more algorithmic techniques to draw from, beyond the QFT and amplitude amplification.

We investigate two such ideas, quantum computation by QA gates design [4]. In this report we are described structure of QFMS and its applications in design technology of intelligent self-organized fuzzy PD-controllers based on Grover's quantum search algorithm model [5]. Main result of quantum computing is exponential (or quadratic) speed-up in comparison to classical computation of problem. In our case we investigate the problem of robust KB design in unpredicted control situations when the classical solution is unknown.

1. Structure of QFMS

Background of developed robust KB design technology is soft computing optimizer (SCO) and QFMS. Structure and functional description of SCO is developed in [6]. We concentrate our attention on QFMS structure.

Figure 1 shows the structure of QFMS.

Figure 2 shows the structure description of the QA Benchmark Block.

Figure 3 shows the general structure of QA.

Quantum algorithms (QA) demonstrate great efficiency in many practical tasks such as factorization of large integer numbers, where classical algorithms are failing or dramatically ineffective [2, 3]. Practical application is still away due to lack of the physical HW implementation of quantum computers. We describe design method of main quantum operators and hardware implementation of QAG for fast search in large database and related topics concerning the control of a process, including search-of-minima *intelligent operations*. This method is very useful for minimum efforts of searching among a set of values and in particular is the first step for the realization of a HW control systems exploiting artificial intelligence in order to fuzzy control in a robust way a non-linear process or in order to efficient search in a database. The presented HW performs all the functional steps of a Grover QSA (This algorithm and its modifications are described in [3]). By suitable changes of traditional matrix approach, a modular n-qubit-hybrid structure is realized in order to prove the usefulness of iterations of the gate, which provide a higher probability of exact solution finding.

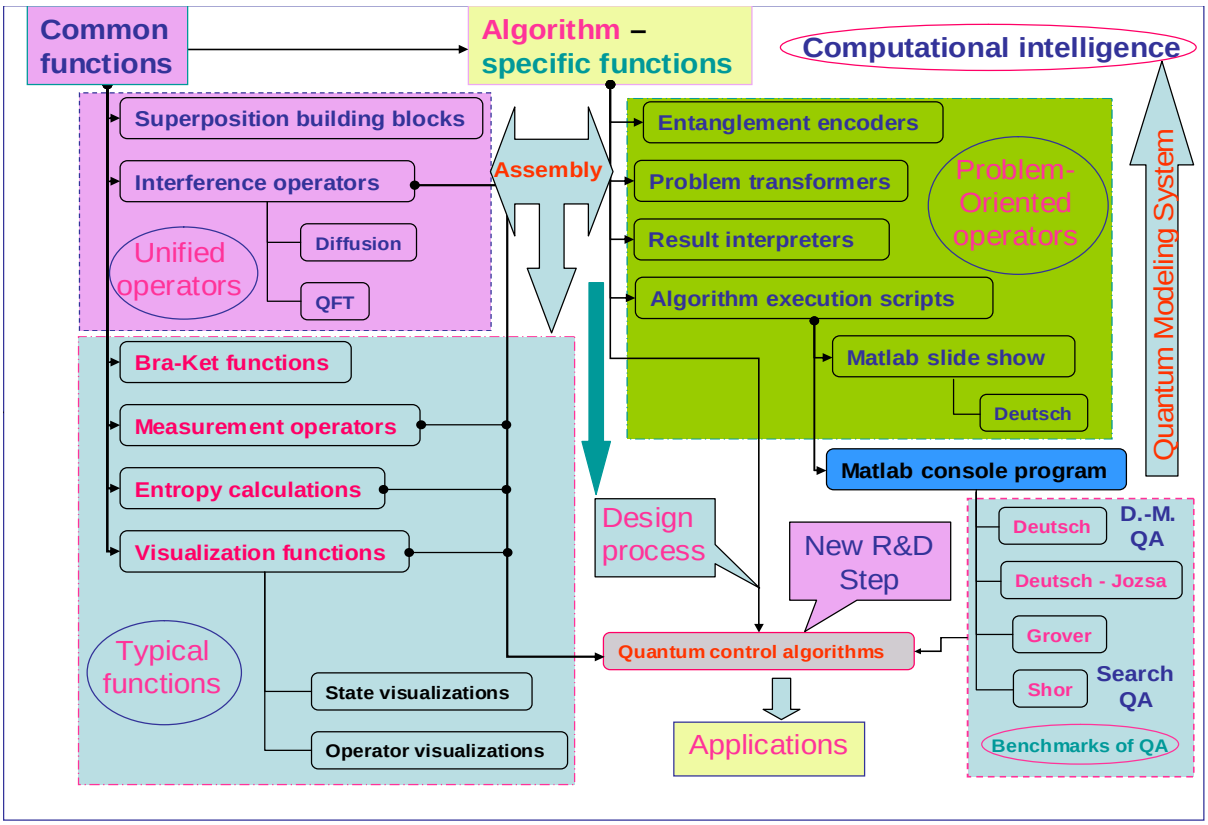

Figure 1: *Structure of QFMS and SW toolkit*

A minimum-entropy based method is adopted as a termination condition criterion and realized in a digital part together with display output.

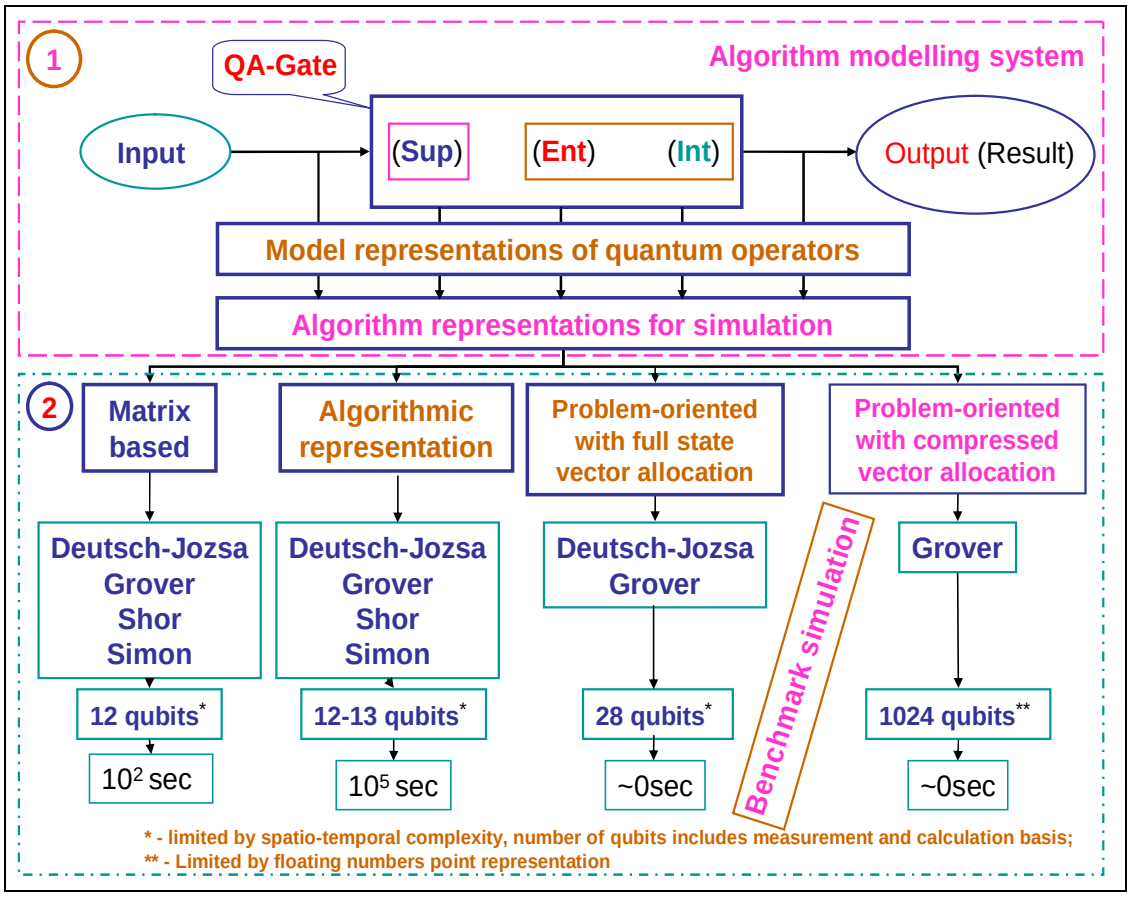

Figure 2: *Algorithm modeling system in QFMS*

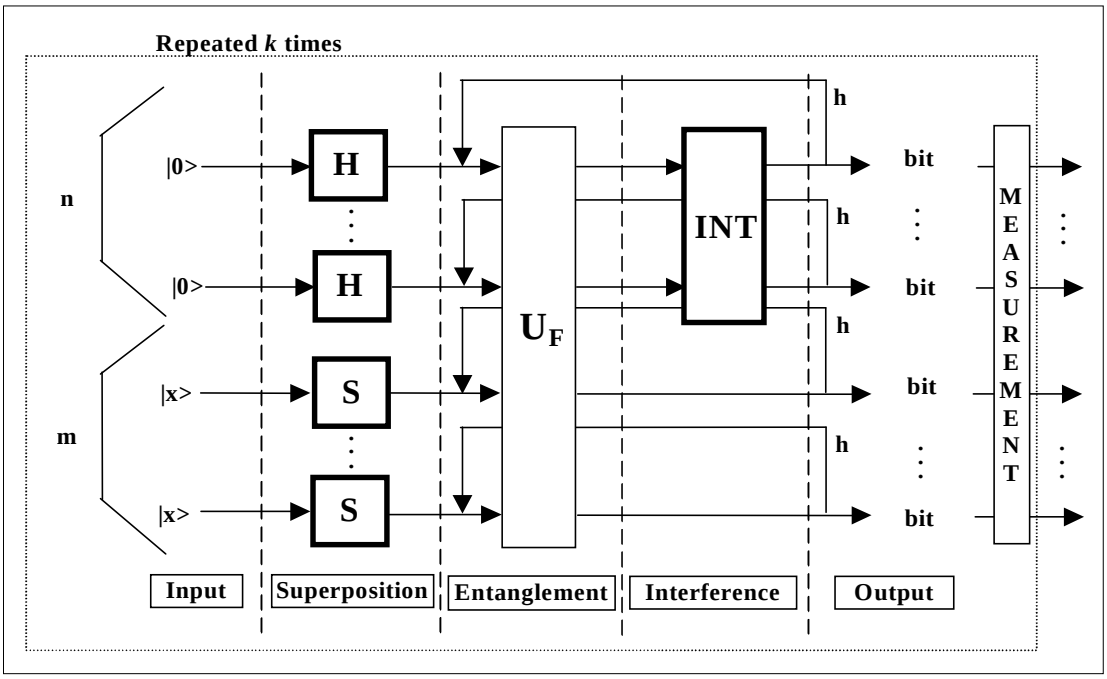

Figure 3: *General structure of quantum algorithm gate*

Figure 4 shows the general approach to application of QFMS.

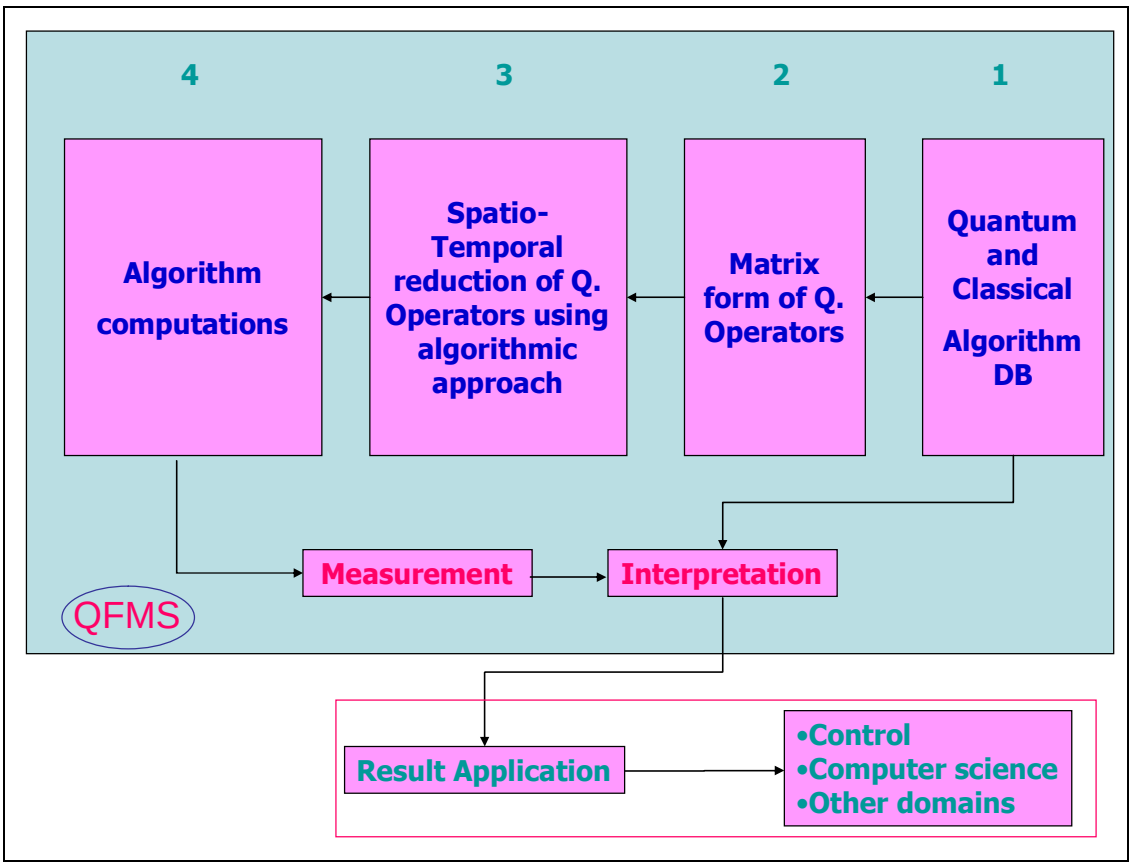

Figure 4: *General approach structure to application of* QFMS

In this modeling system we present *five* practical approaches to design fast algorithms to simulate most of known QAs on classical computers (see, Figure 2):

1. *Matrix based approach*;
2. *Model representations* of quantum operators in fast QAs;

3. *Algorithmic based approach,* when matrix elements are calculated on "demand";
4. *Problem-oriented approach,* where we succeeded to run Grover's algorithm with up to 64 and more qubits with Shannon entropy calculation (up to 1024 without termination condition);
5. *Quantum algorithms with reduced number of operators* (entanglement-free QA, and so on).

Detail description of these approaches is given in [3].

Let us describe briefly the main blocks in Figure 1: (i) unified operators; (ii) problem-oriented operators; (iii) Benchmarks of QA simulation on classical computers; and (iv) quantum control algorithms based on Grover's quantum search algorithm.

2. Description of main QA operators

As we can to see from Figure 3, QA structure has three main quantum operators: superposition; entanglement (quantum oracle); and interference.

We consider superposition, entanglement and interference operators from simulation viewpoint.

In this case superposition and interference have more complicated structure and differ from algorithm to algorithm. And then we consider entanglement operators, since they have similar structure for all QAs, and differ only by function being analyzed.

2.1. *Superposition operators of QA's*. In general, the superposition operator consists of the combination of the tensor products Hadamard H operators with identity operator

$$I:\ H=\frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}, I=\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}.$$

The superposition operator of most QAs can be expressed as (see, Figure 3)

$$Sp=\left(\bigotimes_{i=1}^{n} H\right)\otimes\left(\bigotimes_{i=1}^{m} S\right), \tag{2.1}$$

where n and m are the numbers of inputs and of outputs respectively. Operator, S may be or Hadamard operator H or identity operator I depending on the algorithm.

Numbers of outputs m as well as structures of corresponding superposition and interference operators are presented in the Table 1 for different QAs.

Table 1: *Parameters of superposition and interference operators of main quantum algorithms*

Algorithm	**Superposition**	m	**Interference**
Deutsch's	$H\otimes I$	1	$H\otimes H$
Deutsch-Jozsa's	${}^{n}H\otimes H$	1	${}^{n}H\otimes I$
Grover's	${}^{n}H\otimes H$	1	$D_n\otimes I$
Simon's	${}^{n}H\otimes {}^{n}I$	n	${}^{n}H\otimes {}^{n}I$
Shor's	${}^{n}H\otimes {}^{n}I$	n	$QFT_n\otimes {}^{n}I$

Note that superposition and interference operators are often contain tensor power of Hadamard operator which is called Walsh-Hadamard operator. It is known [7 - 9] that elements of the Walsh-Hadamard operator could be obtained as:

$$\left[{}^{n}H\right]_{i,j}=\frac{(-1)^{i*j}}{2^{n/2}}=\frac{1}{2^{n/2}}\begin{cases}1, & \text{if } i*j \text{ is even}\\ -1, & \text{if } i*j \text{ is odd}\end{cases} \tag{2.2}$$

where $i=0,1,...,2^n$, $j=0,1,...,2^n$. This approach improves greatly performance of classical simulation of the Walsh–Hadamard operators, since its elements could be obtained by the simple replication according to the rule presented in Eq. (2.2).

Example 1: Consider superposition operator of Deutsch's algorithm ($n=1,\ m=1,\ S=I$):

$$[Sp]^{Deutsch}_{i,j} = \frac{(-1)^{i+j}}{2^{1/2}} \otimes I = \frac{1}{\sqrt{2}} \begin{pmatrix} (-1)^{0*0} I & (-1)^{0*1} I \\ (-1)^{1*0} I & (-1)^{1*1} I \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} I & I \\ I & -I \end{bmatrix} \tag{2.3}$$

Example 2: Consider superposition operator of Deutsch-Jozsa's and of Grover's algorithm, for the case ($n = 2,\ \ m = 1,\ \ S = H$):

$$[Sp]^{D.-J.'s, Grover's}_{i,j} = \frac{(-1)^{i+j}}{2^{2/2}} \otimes H = \frac{1}{2} \begin{pmatrix} (-1)^{0*0} H & (-1)^{0*1} H & (-1)^{0*2} H & (-1)^{0*3} H \\ (-1)^{1*0} H & (-1)^{1*1} H & (-1)^{1*2} H & (-1)^{1*3} H \\ (-1)^{2*0} H & (-1)^{2*1} H & (-1)^{2*2} H & (-1)^{2*3} H \\ (-1)^{3*0} H & (-1)^{3*1} H & (-1)^{3*2} H & (-1)^{3*3} H \end{pmatrix}$$

$$= \frac{1}{2} \begin{pmatrix} H & H & H & H \\ H & -H & H & -H \\ H & H & H & H \\ H & -H & H & -H \end{pmatrix} \tag{2.4}$$

Example 3: Superposition operator of Simon's (and Shor's) algorithms ($n = 2, m = 2, S = I$):

$$[Sp]^{Simon, Shor}_{i,j} = \frac{(-1)^{i+j}}{2^{2/2}} \otimes^2 I = \frac{1}{2} \begin{pmatrix} (-1)^{0*0} ({}^2 I) & (-1)^{0*1} ({}^2 I) & (-1)^{0*2} ({}^2 I) & (-1)^{0*3} ({}^2 I) \\ (-1)^{1*0} ({}^2 I) & (-1)^{1*1} ({}^2 I) & (-1)^{1*2} ({}^2 I) & (-1)^{1*3} ({}^2 I) \\ (-1)^{2*0} ({}^2 I) & (-1)^{2*1} ({}^2 I) & (-1)^{2*2} ({}^2 I) & (-1)^{2*3} ({}^2 I) \\ (-1)^{3*0} ({}^2 I) & (-1)^{3*1} ({}^2 I) & (-1)^{3*2} ({}^2 I) & (-1)^{3*3} ({}^2 I) \end{pmatrix} \tag{2.5}$$

$$= \frac{1}{2} \begin{pmatrix} {}^2 I & {}^2 I & {}^2 I & {}^2 I \\ {}^2 I & -{}^2 I & {}^2 I & -{}^2 I \\ {}^2 I & {}^2 I & {}^2 I & {}^2 I \\ {}^2 I & -{}^2 I & {}^2 I & -{}^2 I \end{pmatrix}$$

2.2. *Interference operators of main QA's* Interference operators must be selected for each algorithm individually according to the parameters presented in the Table 1. Consider some particular parts of interference operators. Interference operator consists of interference part, which is different for all algorithms, and from measurement part, which is the same for most of algorithms and consists of m tensor power of identity operator.

Consider interference operator of each algorithm.

Example 1: *Interference operator of Deutsch' algorithm*. Interference operator of Deutsch's algorithm consists of tensor product of two Hadamard transformations, and can be calculated using Eq. (2.4) with $n = 2$:

$$\left[Int^{\ Deutsch} \right]_{i,j} = {}^2 H = \frac{(-1)^{\ i*j}}{2^{2/2}} = \frac{1}{2} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \end{pmatrix} \tag{2.8}$$

Note that in Deutsch's algorithm, Walsh-Hadamard transformation in interference operator is used also for the measurement basis.

Example 2: *Interference operator of Deutsch-Jozsa's algorithm*. Interference operator of Deutsch-Jozsa's algorithm consists of tensor product of n power of Walsh-Hadamard operator with an identity operator. In general form the block matrix of the interference operator of Deutsch-Jozsa's algorithm can be written as:

$$\left[Int^{\ Deutsch-Jozsa's} \right]_{i,j} = \frac{(-1)^{\ i*j}}{2^{\frac{n}{2}}} \otimes I\ , \tag{2.9}$$

where $i = 0,...,2^n - 1,\ j = 0,...,2^n - 1$.

Example 3: *Interference operator of Deutsch-Jozsa's algorithm* ($n = 3, k_1 = 2, k_2 = 1$):

$$\left[Int^{Deutsch-Jozsa's}\right]_{i,j} = \frac{(-1)^{i*j}}{2^{\frac{2}{2}}} \otimes I = \frac{1}{2}\begin{pmatrix} I & I & I & I \\ I & -I & I & -I \\ I & I & I & I \\ I & -I & I & -I \end{pmatrix} \tag{2.10}$$

Example 4: *Interference operator of Grover's algorithm*. Interference operator of Grover's algorithm can be written as a block matrix of the following form:

$$\left[Int^{Grover}\right]_{i,j} = D_n \otimes I = \left(\frac{1}{2^{n/2}} - {}^{n}I\right) \otimes I = \left(-1 + \frac{1}{2^{n/2}}\right) \otimes I \Big|_{i=j}, \left(\frac{1}{2^{n/2}}\right) \otimes I \Big|_{i \neq j} = \frac{1}{2^{n/2}} \begin{cases} -I, i = j \\ I, i \neq j \end{cases} \tag{2.11}$$

where $i = 0,...,2^n - 1,\ j = 0,...,2^n - 1$, D_n refers to diffusion operator: $\left[D_n\right]_{i,j} = \frac{(-1)^{1AND(i=j)}}{2^{n/2}}$.

Example 5: *Interference operator of Grover's QSA* ($n = 2, m = 1$):

$$\left[Int^{Grover}\right]_{i,j} = D_2 \otimes I = \left(\frac{1}{2^{2/2}} - {}^{2}I\right) \otimes I = \left(-1 + \frac{1}{2}\right) \otimes I \Big|_{i=j}, \left(\frac{1}{2}\right) \otimes I \Big|_{i \neq j}$$

$$= \begin{pmatrix} \left(-1+\frac{1}{2}\right)I & \frac{1}{2}I & \frac{1}{2}I & \frac{1}{2}I \\ \frac{1}{2}I & \left(-1+\frac{1}{2}\right)I & \frac{1}{2}I & \frac{1}{2}I \\ \frac{1}{2}I & \frac{1}{2}I & \left(-1+\frac{1}{2}\right)I & \frac{1}{2}I \\ \frac{1}{2}I & \frac{1}{2}I & \frac{1}{2}I & \left(-1+\frac{1}{2}\right)I \end{pmatrix} = \frac{1}{2}\begin{pmatrix} -I & I & I & I \\ I & -I & I & I \\ I & I & -I & I \\ I & I & I & -I \end{pmatrix} \tag{2.12}$$

Note that with bigger number of qubits, gain coefficient will become smaller. Dimension of the matrix increases according to 2^n, but each element can be extracted using Eq. (2.11), without allocation of entire operator matrix.

Remark. Since $D_n D_n^* = I$, D_n is unitary and is therefore a possible quantum state transformation. While the matrix D_n is clearly unitary it can to have the decomposition form $D_n = -H_n R_n^1 H_n$, where $R_n^1[i,j] = 0$, if $i \neq j$, $R_1^1[1,1] = -1$ and $R_n^1[i,i] = +1$, if $1 < i \leq N$. In concrete form the operator D_n (*diffusion – inversion about average*) in Grover algorithm is decomposed as

$$\boxed{D_n = \frac{1}{\sqrt{2^n}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}^{\otimes n} \cdot \begin{pmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}^{\otimes n} \cdot \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}^{\otimes n}}$$

and can be accomplished with $O(n) = O(\log(N))$ quantum gates [3]. It means that from the viewpoint of efficient computation the form as in Eq. (2.11) is more preferable.

Example 6: *Interference operator of Simon's algorithm*. Interference operator of Simon's algorithm is prepared in the same manner as superposition as well as superposition operators of Shor's algorithms and can be described as following Eq. (2.3) and Eq. (2.7)

$$\left[Int^{Simon}\right]_{i,j} = {}^{n}H \otimes {}^{m}I = \frac{(-1)^{i*j}}{2^{n/2}} \otimes {}^{m}I = \frac{1}{2^{n/2}}\begin{pmatrix} (-1)^{0*0} \cdot {}^{m}I & \cdots & (-1)^{0*j} \cdot {}^{m}I & \cdots & (-1)^{0*(2^n-1)} \cdot {}^{m}I \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ (-1)^{i*0} \cdot {}^{m}I & \cdots & (-1)^{i*j} \cdot {}^{m}I & \cdots & (-1)^{i*(2^n-1)} \cdot {}^{m}I \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ (-1)^{(2^n-1)*0} \cdot {}^{m}I & \cdots & (-1)^{(2^n-1)*j} \cdot {}^{m}I & \cdots & (-1)^{(2^n-1)*(2^n-1)} \cdot {}^{m}I \end{pmatrix} \tag{2.13}$$

Remark. In general, interference operator of Simon's algorithm coincides with interference operator of Deutsch-Jozsa's algorithm Eq. (2.9), but each block of the operator matrix Eq. (2.13) consists of m tensor products of identity operator.

Remark. Each odd block (when product of the indexes is an odd number) of the Simon's interference operator Eq. (2.13), has a negative sign. Actually if $i = 0, 2, 4, \ldots 2^n - 2$ or $j = 0, 2, 4, \ldots 2^n - 2$ the block sign is positive, else block sign is negative. This rule is applicable also for Eq. (2.9) of Deutsch-Jozsa's algorithm interference operator. Then it is convenient to check if one of the indexes is an even number instead of calculating their product. Then Eq. (2.13) can be reduced as:

$$\left[Int^{Simon}\right]_{i,j} = {}^{n}H \otimes {}^{m}I = \frac{(-1)^{i*j}}{2^{n/2}} \otimes {}^{m}I = \frac{1}{2^{n/2}} \begin{cases} {}^{m}I, \text{if } i \text{ is odd or if } j \text{ is odd} \\ -{}^{m}I, \text{if } i \text{ is even and } j \text{ is even} \end{cases} \tag{2.13a}$$

Example 7: *Interference operator of Shor's algorithm.* Interference operator of Shor's algorithm uses Quantum Fourier Transformation (QFT) operator, calculated as:

$$\left[QFT_n\right]_{i,j} = \frac{1}{2^{n/2}} e^{J(i*j)\frac{2\pi}{2^n}} \tag{2.14}$$

where: J - imaginary unit, $i = 0, \ldots, 2^n - 1$ and, $j = 0, \ldots, 2^n - 1$. With $n = 1$ we can observe the following relation:

$$QFT_{k_1}\Big|_{k_1=1} = \frac{1}{2^{\frac{1}{2}}} \begin{pmatrix} e^{J*(0*0)2\pi/2^1} & e^{J*(0*1)2\pi/2^1} \\ e^{J*(1*0)2\pi/2^1} & e^{J*(1*1)2\pi/2^1} \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} = H \tag{2.15}$$

Eq. (2.15) can be also presented in harmonic form using Euler formula:

$$\left[QFT_{k_1}\right]_{i,j} = \frac{1}{2^{k_1/2}} \left(\cos\left((i*j)\frac{2\pi}{2^{k_1}} \right) + J \sin\left((i*j)\frac{2\pi}{2^{k_1}} \right) \right)$$

2.3. *Entanglement operators of main QA's* In general entanglement operators are part of QA where the information about the function being analyzed is coded as input-output relation. Let's discuss the general approach for coding binary functions into corresponding entanglement gates. Consider arbitrary binary function:

$$f : \{0,1\}^n \to \{0,1\}^m, \text{ such that } f(x_0, \ldots, x_{n-1}) = (y_0, \ldots, y_{m-1}).$$

In order to create unitary quantum operator, which performs the same transformation, first we transfer irreversible function f into reversible function F, as following:

$$F : \{0,1\}^{m+n} \to \{0,1\}^{m+n},$$

such that

$$F\left(x_0, \ldots, x_{n-1}, y_0, \ldots, y_{m-1}\right) = (x_0, \ldots, x_{n-1}, f(x_0, \ldots, x_{n-1}) \oplus (y_0, \ldots, y_{m-1}))$$

where $\oplus$ denotes addition modulo 2. Having reversible function F we can design an entanglement operator matrix using the following rule:

$$\left[U_F\right]_{i^B, j^B} = 1 \text{ iff } F(j^B) = i^B, \quad i, j \in \left[\underbrace{0,..,0}_{n+m};\underbrace{1,..,1}_{n+m};\right] \quad B \text{ denotes binary coding}$$

Actually resulted entanglement operator is a block diagonal matrix, of the form:

$$U_F = \begin{pmatrix} M_0 & & 0 \\ & \ddots & \\ 0 & & M_{2^n-1} \end{pmatrix}$$

Each block $M_i, i = 0, \ldots, 2^n - 1$ consists of m tensor products of I or of C operators, and can be obtained as following:

$$M_i = \bigotimes_{k=0}^{m-1} \begin{cases} I, \text{iff } F(i,k) = 0 \\ C, \text{iff } F(i,k) = 1 \end{cases}, \tag{2.16}$$

where C stays for NOT operator, defined as:

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

It is clear that entanglement operator is a *sparse* matrix. Using sparse matrix operations it is possible to accelerate the simulation of entanglement operation.

Example 1. Entanglement operator for binary function: $f:\{0,1\}^2 \to \{0,1\}^1$ such that:

$$f(x) = 1|_{x=01}\, 0|_{x\neq 01}.$$

Reversible function F in this case will be:

$$F:\{0,1\}^3 \to \{0,1\}^3,$$

such that:

(x, y)	$(x, f(x) \oplus y)$
$00,0$	$00,0 \oplus 0 = 0$
$00,1$	$00,0 \oplus 1 = 1$
$01,0$	$01,1 \oplus 0 = 1$
$01,1$	$01,1 \oplus 1 = 0$
$10,0$	$10,0 \oplus 0 = 0$
$10,1$	$10,1 \oplus 0 = 1$
$11,0$	$11,0 \oplus 0 = 0$
$11,1$	$11,1 \oplus 0 = 1$

And corresponding entanglement block matrix can be written as:

$$U_F = \begin{array}{c} \\ |00\rangle \\ |01\rangle \\ |10\rangle \\ |11\rangle \end{array} \begin{array}{c} \begin{array}{cccc} \langle 00| & \langle 01| & \langle 10| & \langle 11| \end{array} \\ \begin{pmatrix} I & 0 & 0 & 0 \\ 0 & \boxed{C} & 0 & 0 \\ 0 & 0 & I & 0 \\ 0 & 0 & 0 & I \end{pmatrix} \end{array} \tag{2.17}$$

Eq. (2.17) demonstrates the result of the application of this operator in Grover's QSA. Entanglement operators of Deutsch and of Deutsch-Jozsa's algorithms have the same form.

Example 2: Entanglement operator for binary function: $f:\{0,1\}^2 \to \{0,1\}^2$, such that $f(x) = 10|_{x=01,11}\, 00|_{x\neq 01,11}$ and

$$U_F = \begin{array}{c} \\ |00\rangle \\ |01\rangle \\ |10\rangle \\ |11\rangle \end{array} \begin{array}{c} \begin{array}{cccc} \langle 00| & \langle 01| & \langle 10| & \langle 11| \end{array} \\ \begin{pmatrix} I \otimes I & 0 & 0 & 0 \\ 0 & \boxed{C \otimes I} & 0 & 0 \\ 0 & 0 & I \otimes I & 0 \\ 0 & 0 & 0 & \boxed{C \otimes I} \end{pmatrix} \end{array} \tag{2.18}$$

Entanglement operators of Shor's and of Simon's algorithms have the same form.

3. SW&HW support of QA computing accelerator

. Figure 5 shows the structure of intelligent quantum computing accelerator. This structure is developed for the realization of QA and quantum operators. HW of quantum computing accelerator is based on standard silicon element background [7 - 9].

Figure 6a shows the QA structure for HW and MatLab (Figure 6b) implementations.

Termination condition for QA iteration stopping is realized on criteria of minimum Shannon entropy.

Figure 7 shows digital block of Shannon entropy minimum calculation and the main idea of the termination criterion based on this minimum of entropy.

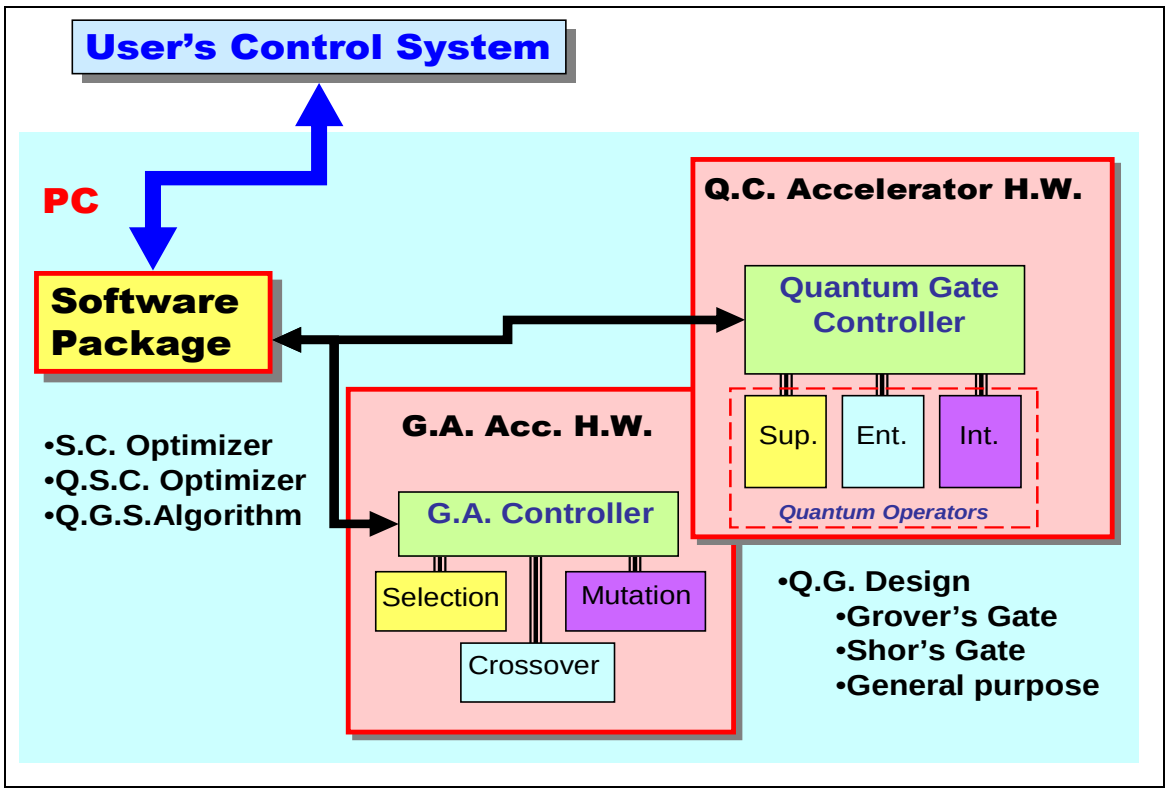

Figure 5: *Structure of intelligent quantum computing accelerator*

Number of iterations of QA is defined during the calculation process of minimum entropy search. In this case QA with minimum of entropy is called *intelligent* QA.

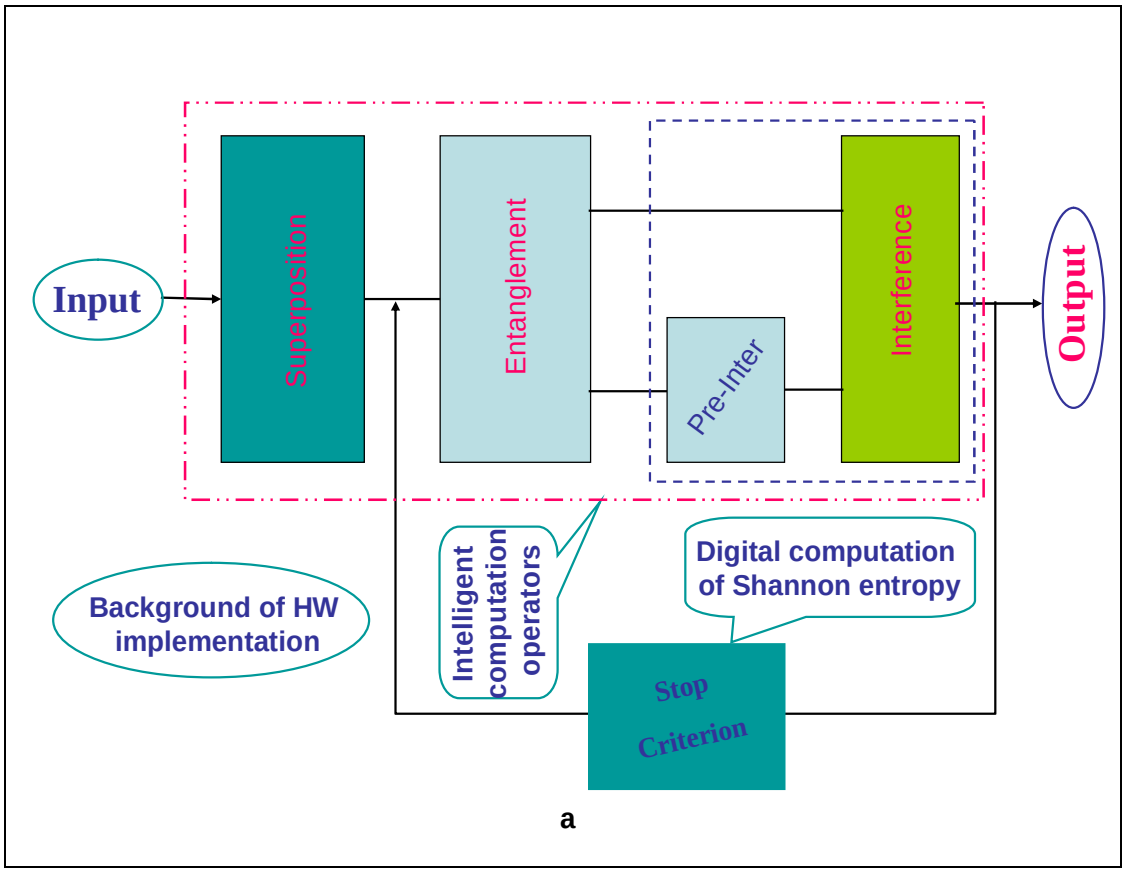

Figure 6: *QA structure presentation for HW* (a)

and

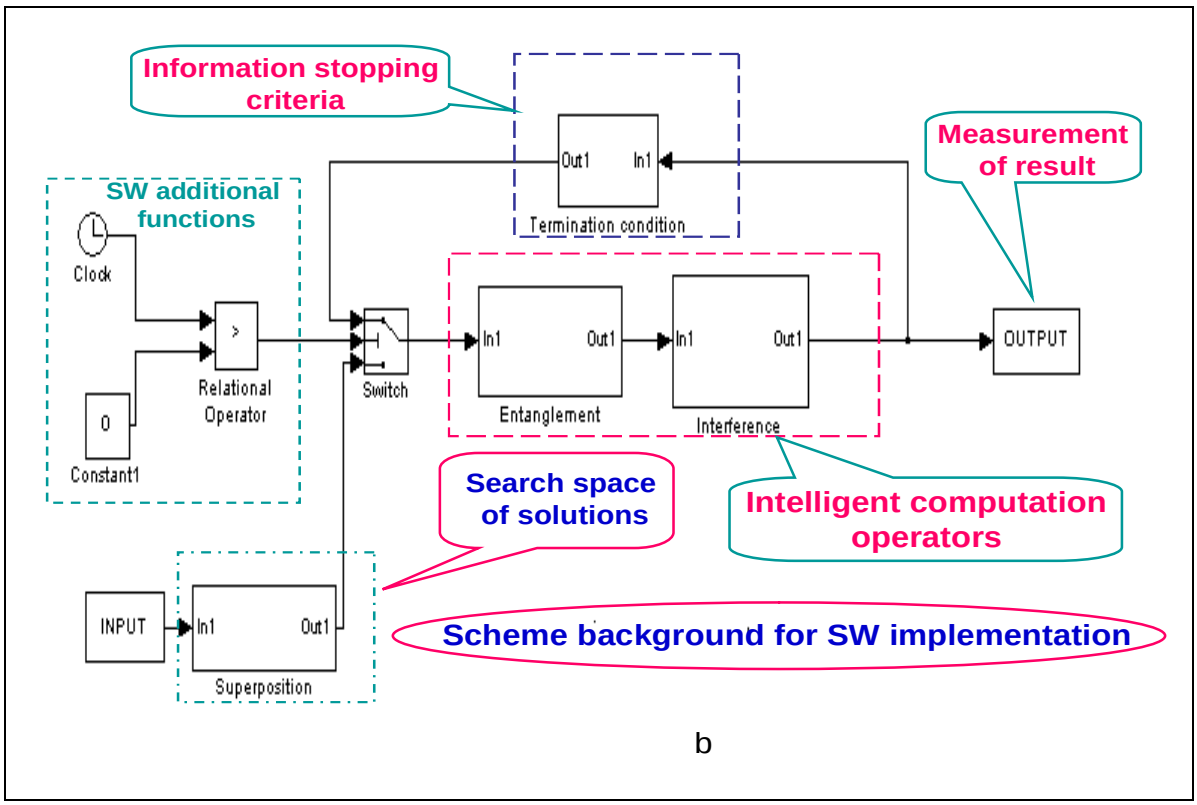

Figure 6: MatLab (b) *implementations*

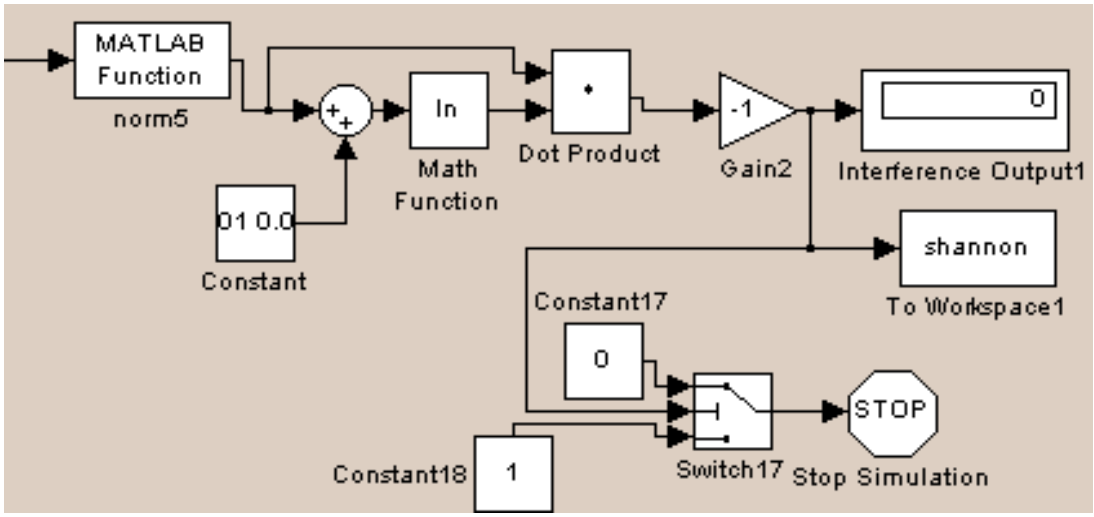

Figure 7: *Digital block of Shannon entropy minimum calculation*

Let us consider briefly the structure of HW implementation of main quantum operators: superposition, interference and entanglement [7 - 9].

Figure 8 shows the structure of superposition and interference operator simulation.

Figure 9 shows the superposition modeling circuit.

According to the rules of quantum computing

$$[1 \quad -1]^{\mathrm{T}} = \begin{bmatrix} 1 \\ -1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ -1 \end{bmatrix} = \left(|0\rangle - |1\rangle\right),$$

i.e., we have the superposition state.

Figure 10 shows of qubits simulation circuits with tensor product.

Figure 11 shows the computation of entanglement operators.

Figure 12 shows the entanglement creation circuit.

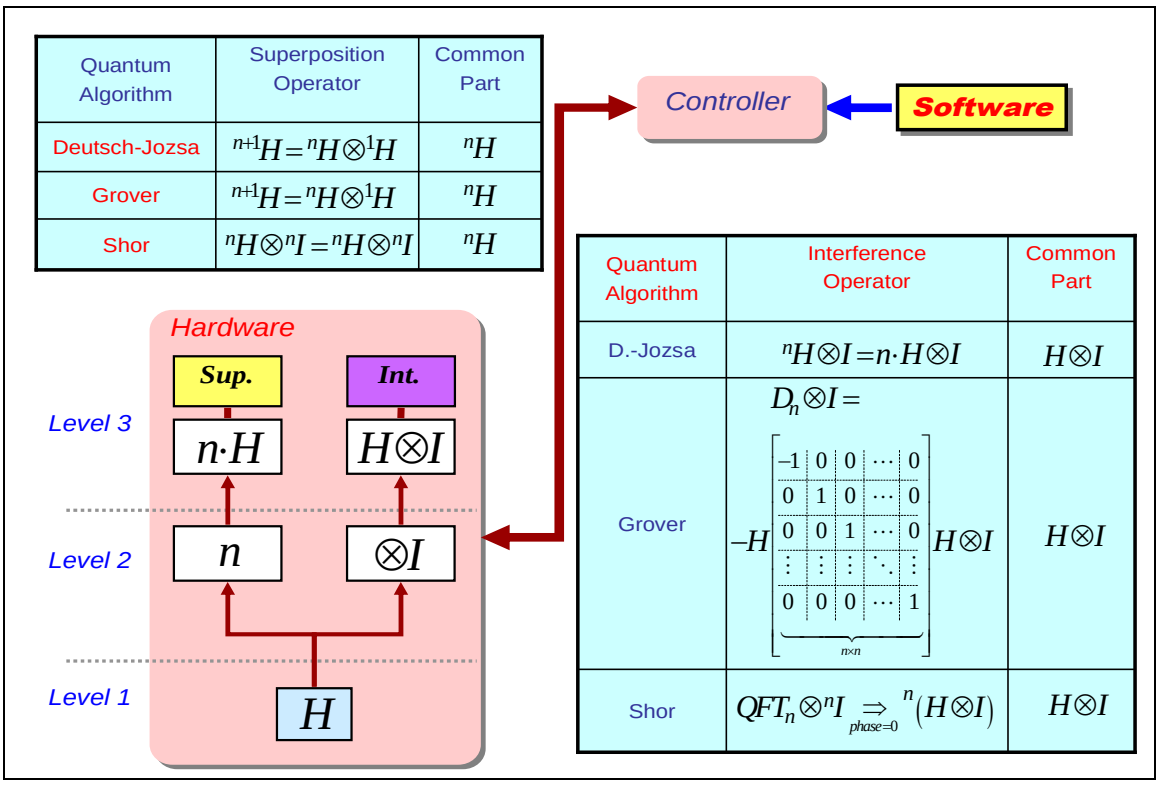

Figure 8: *Computation of superposition and interference operators*

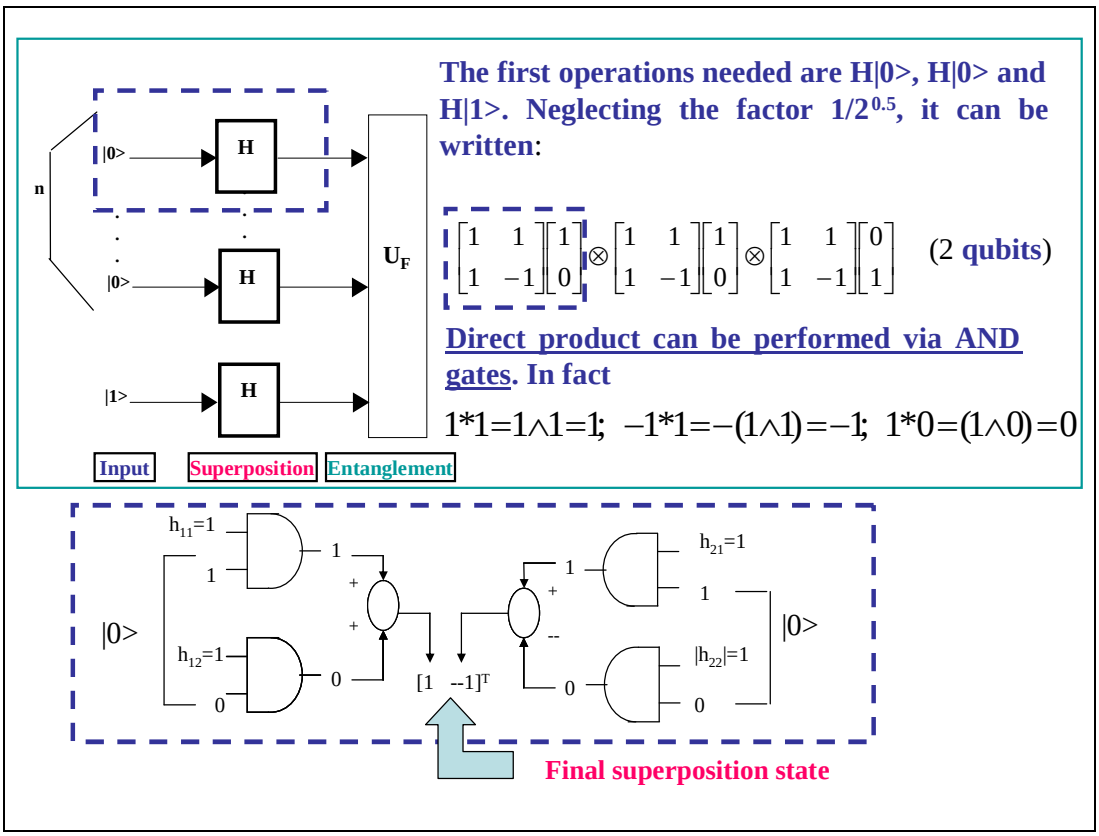

Figure 9: *Superposition modeling circuit*

Another comment relates to the particular form of superposition that have nonzero element in predictable position. This means that we can obtain output of Entanglement $G=U_F \cdot Y$ without calculate matrix product, but only having knowledge of corresponding row of diagonal U_F matrix (see below, Figure 13).

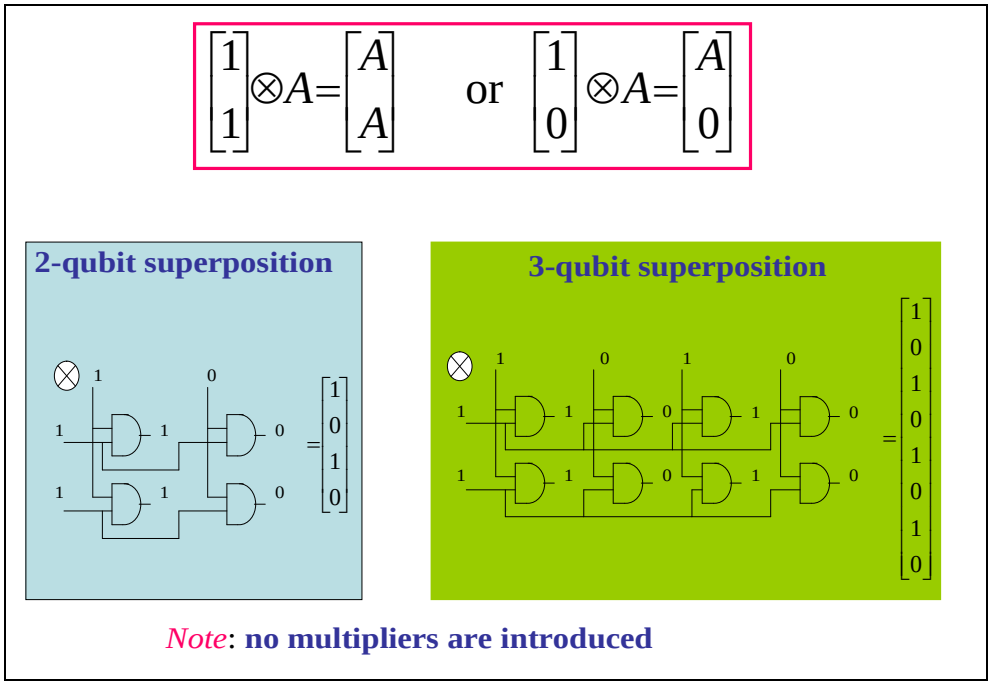

Figure 10: *Qubits simulation circuits with tensor product*

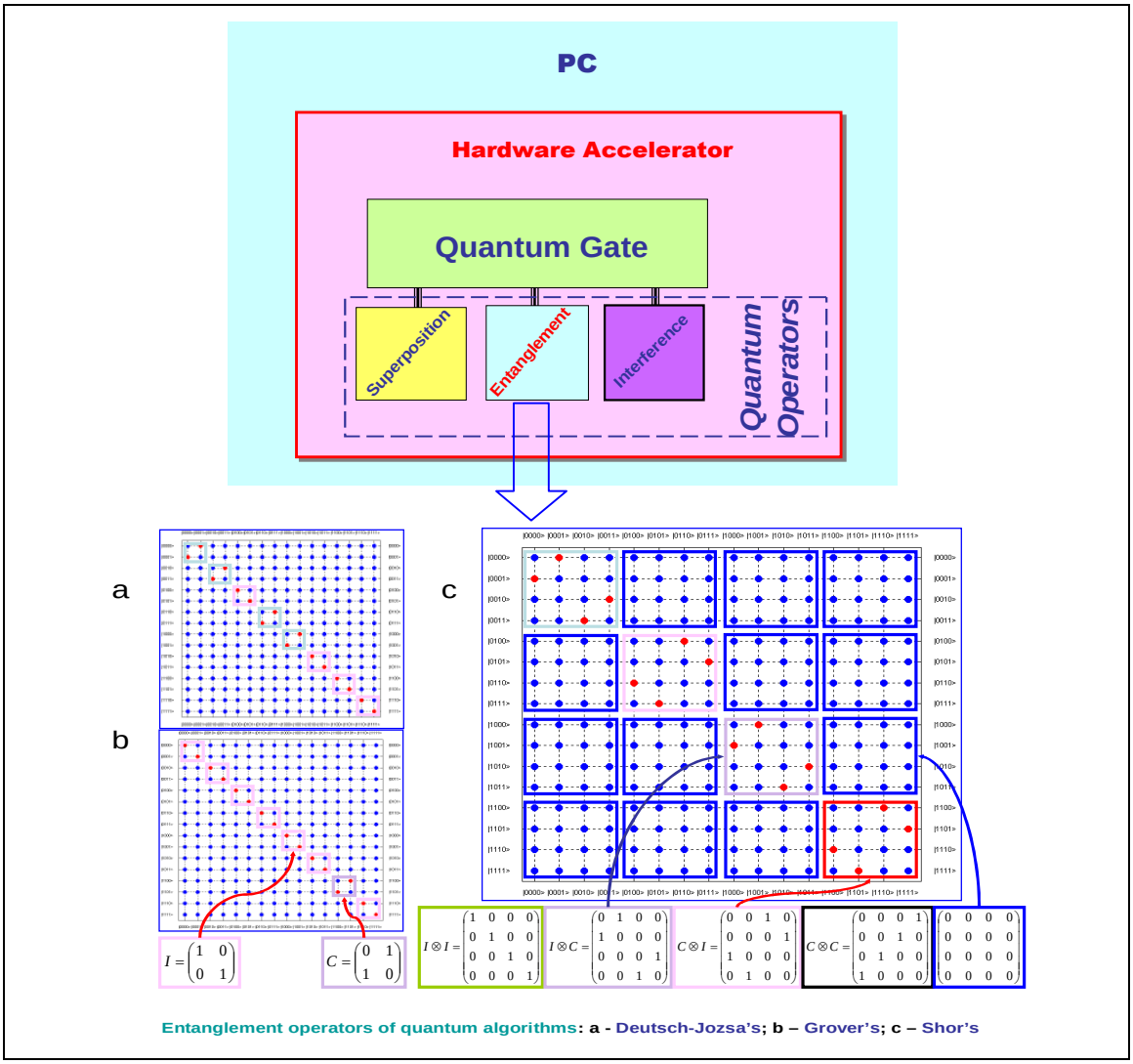

Figure 11: *The computation of entanglement operators*

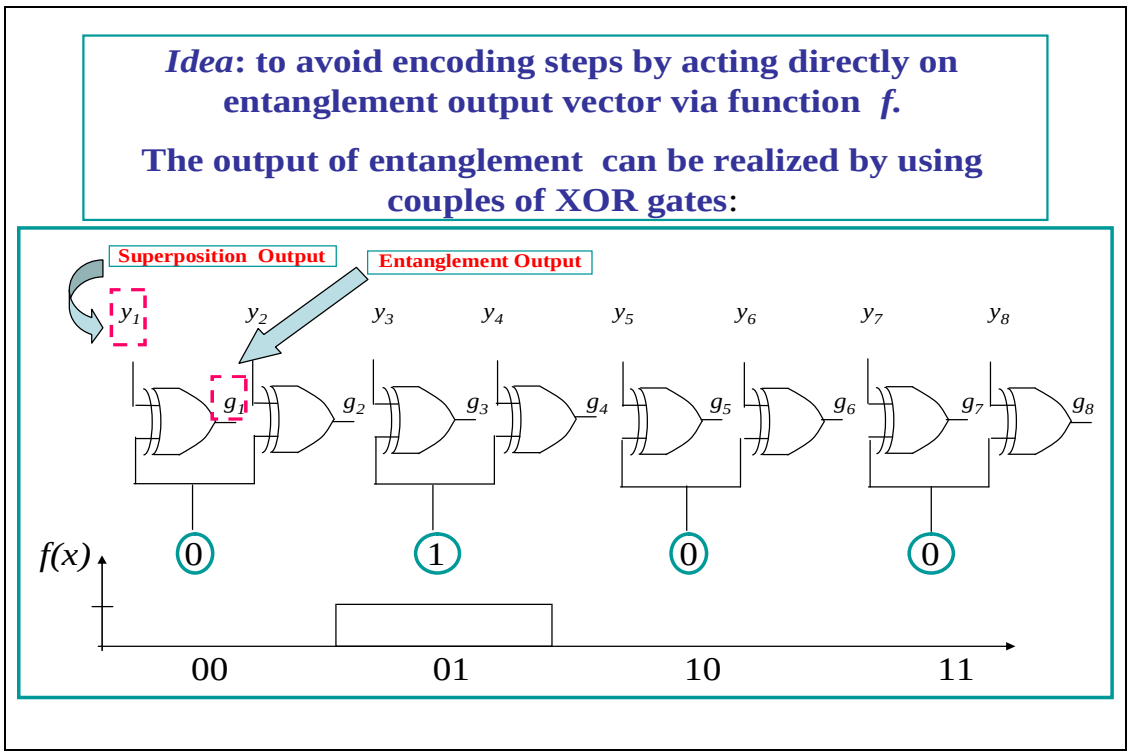

Figure 12: *The entanglement creation circuit*

More in detail we observe that only first row of each $2^n \times 2^n$ block of entanglement contribute to this output vector meaning a strong reduction of computation complexity. In addition we can easily calculate this rows that have the only nonzero element of each block in position $f(x_j)$ +1 (see, Figure 12). Finally we can write output vector *G* as following (Figure 13, *Shor*'s QA):

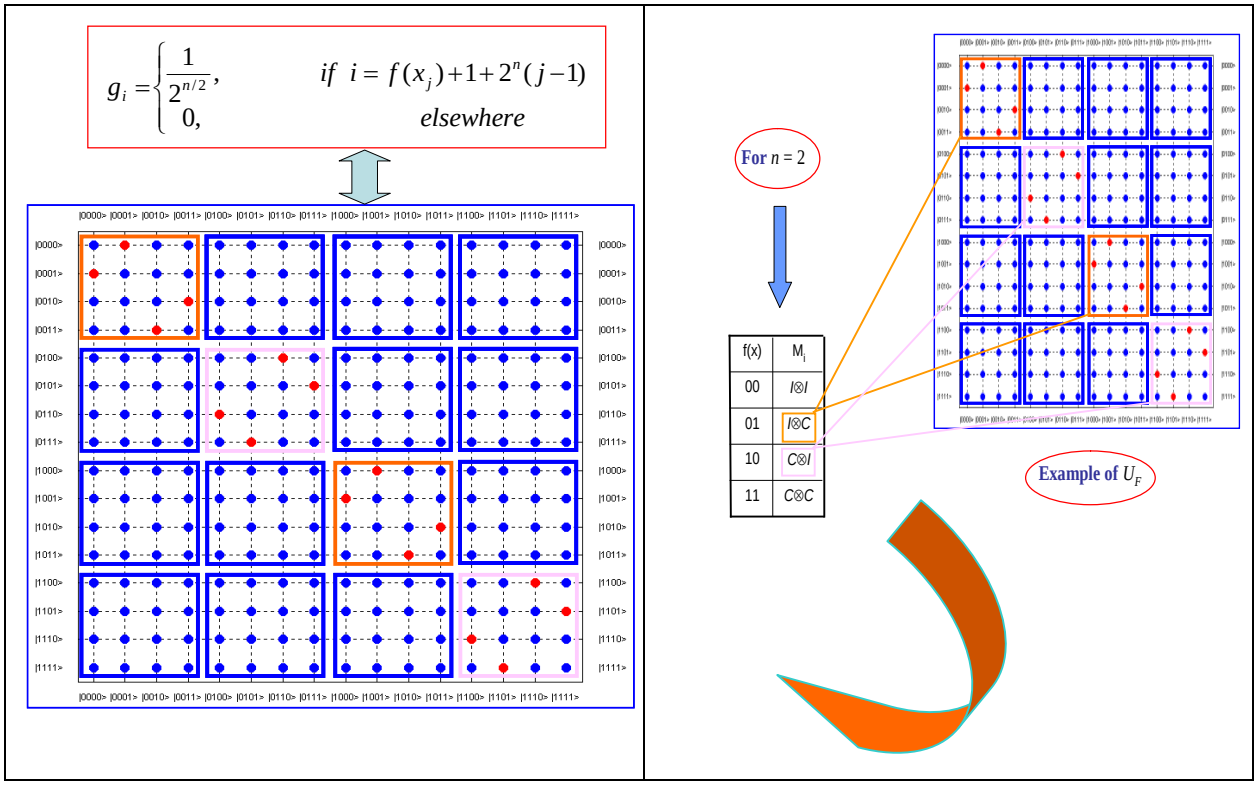

Figure 13: *Equivalent form of output vector G*

Figure 14 shows the entanglement circuit realization.

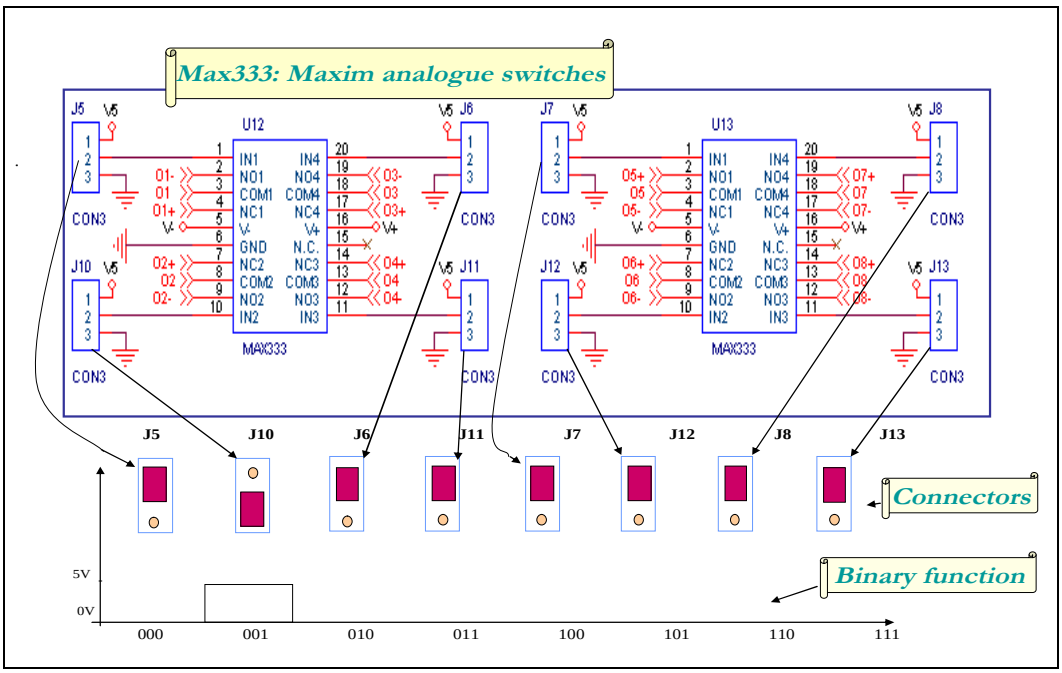

Figure 14: *Entanglement circuit realization*

Figure 15 shows the circuit realization of interference operator according to the scheme in Figure 6a.

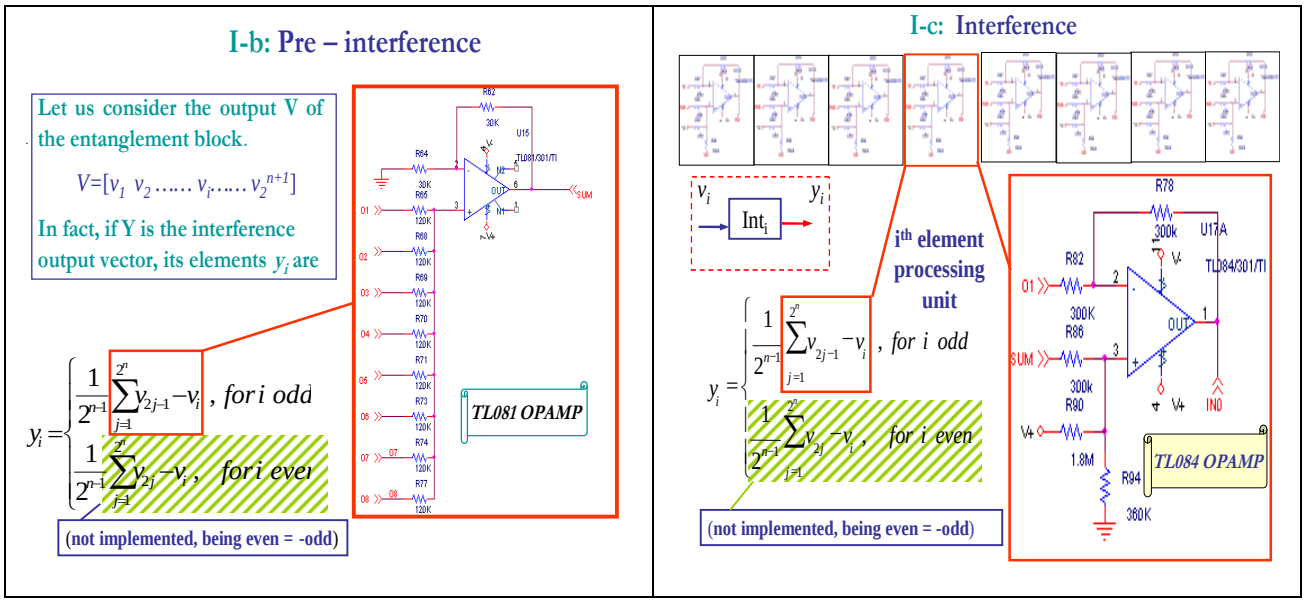

Figure 15: *Interference circuit realization*

Remark. As previously reported [7, 8], in Grover's algorithm the gate is prepared with first n qubits set to $|0\rangle$ and qubit $n+1$ set to $|1\rangle$. Since superposition block is constituted by $H \otimes H \otimes \ldots \otimes H = {}^{n}H$, the output vector Y can be represented in the following way:

$$Y = \left[y_1, y_2, \ldots, y_i, y_{2^{n+1}} \right]$$

where $y_i = \dfrac{(-1)^{i+1}}{2^{\frac{n+1}{2}}}$. Different consideration has to be done for Shor's algorithm. In fact, even if all of $2n$ qubits are more easily set to |0>.

In this case superposition block is ${}^{n}H \otimes {}^{n}I$:

$$y_i = \begin{cases} \dfrac{1}{2^{\frac{n}{2}}}, & \text{if } \; i = 1 + 2^n (j-1) \\ 0, & \text{elsewhere} \end{cases}$$

with $\quad j = 1, \ldots, 2^n; \quad i = 1, \ldots, 2^{2n}$.

Let us consider effectiveness of QFMS with standard QA Benchmarks.

Example 1: *Grover*'s *quantum search algorithm*. Figure 16 shows the structure of Grover's QA gate.

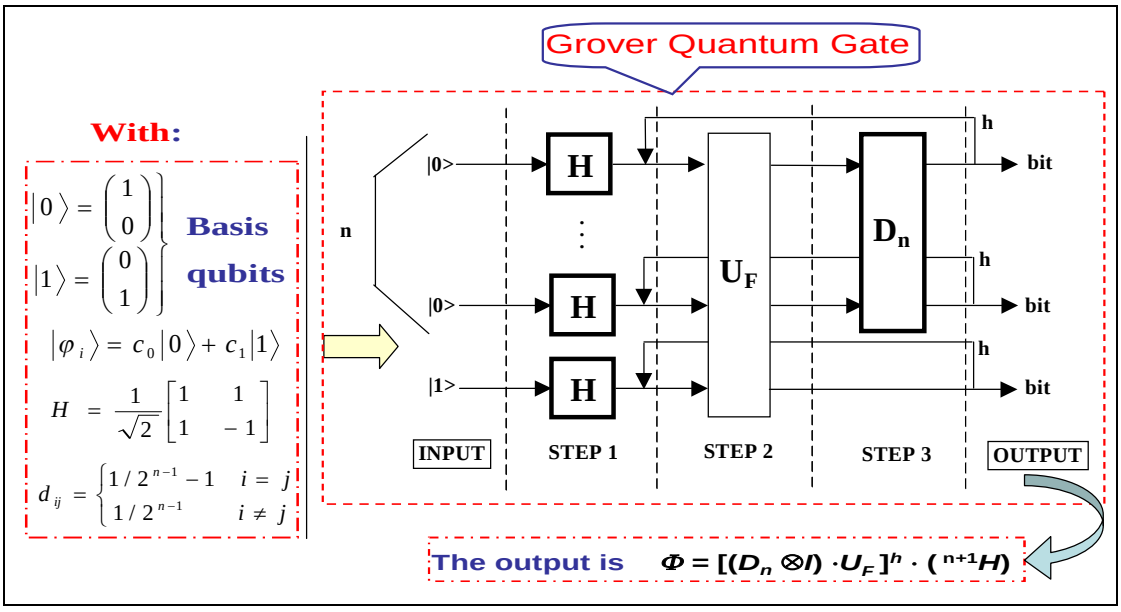

Figure 16: *Grover's QA gate*

Figure 17 shows the Grover's gate circuit.

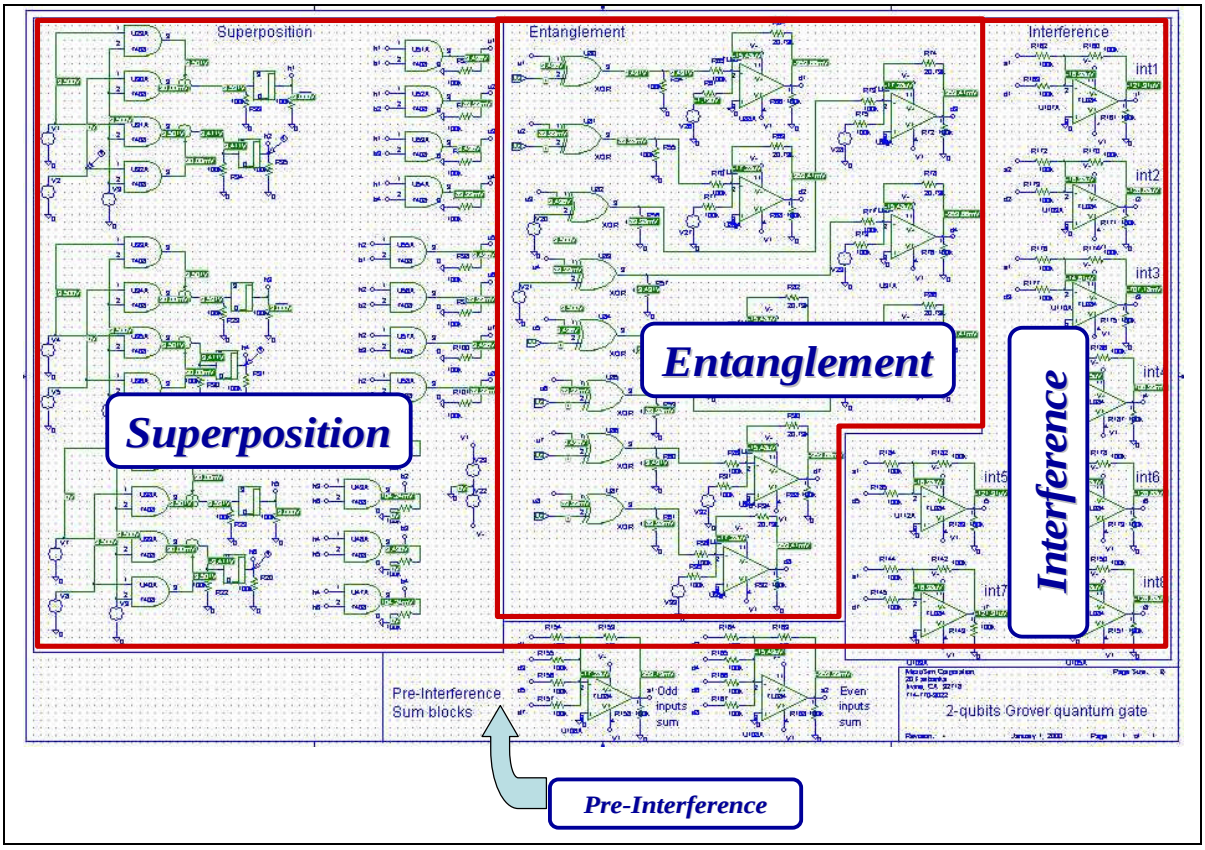

Figure 17: *Quantum Grover gate circuit* [2 qubit PSPICE model (No iteration)]

This schematic realizes superposition, entanglement and interference blocks of a Grover's quantum gate.

Figure 18 shows actual evolution of Grover's quantum search algorithm for three qubits.

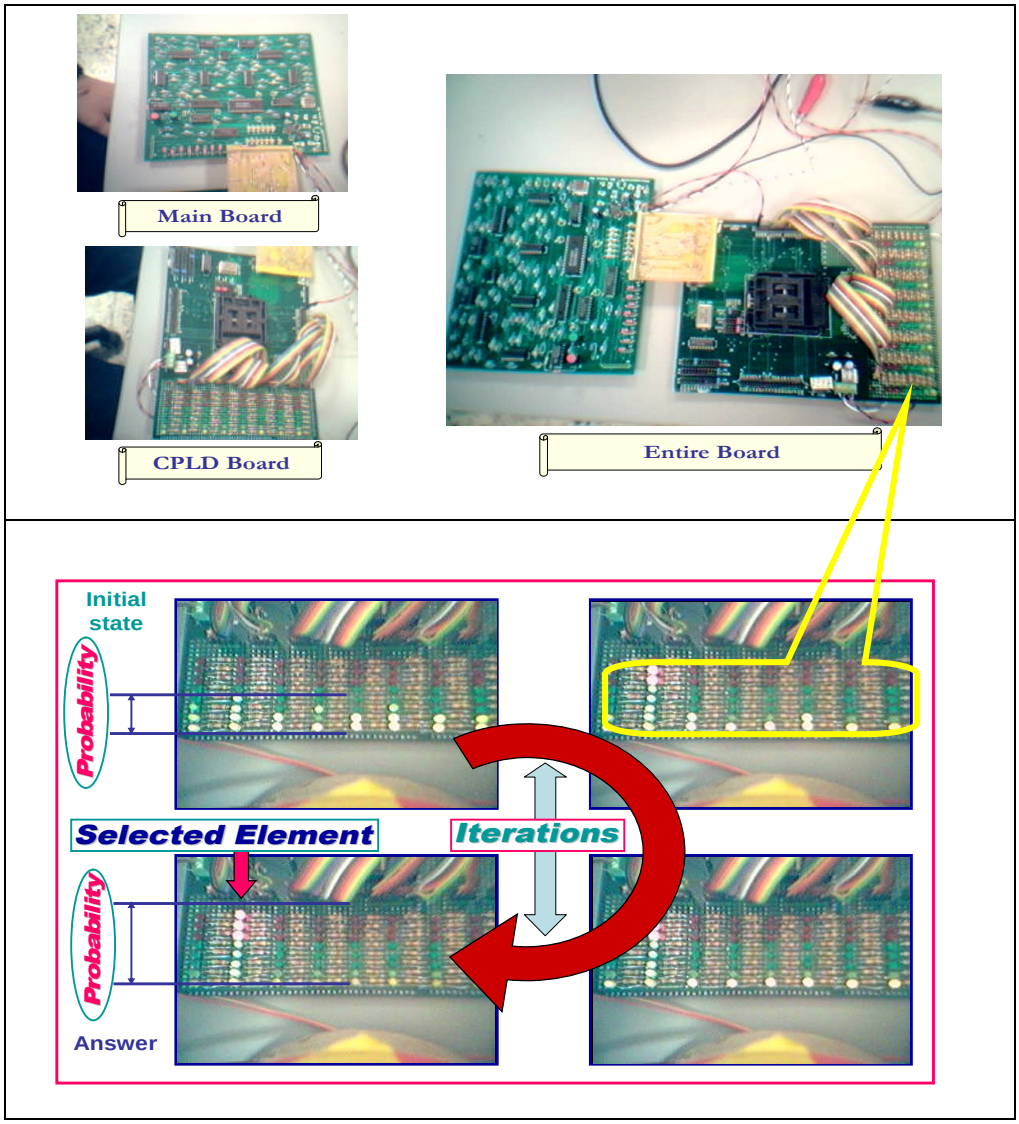

Figure 18: *Actual evolution of Grover's quantum search algorithm for three qubits*

The *SW-system* is divided into two general sections (see, Figure 1).

The first section involves common functions. The second section involves algorithm-specific functions for realizing the concrete algorithms.

Common functions The common functions include:

- Superposition building blocks;
- Interference building blocks;
- Bra-Ket functions;
- Measurement operators;
- Entropy calculation operators;
- Visualization functions;
- State visualization functions;
- Operator visualization functions

Algorithm-specific functions. The algorithm-specific functions include:

- Entanglement encoders;
- Problem transformers;

- Result interpreters;
- Algorithm execution scripts;
- Deutsch algorithm execution script;
- Deutsch Jozsa's algorithm execution script;
- Grover's algorithm execution script;
- Shor's algorithm execution script;
- Quantum control algorithms based on QFI as scripts

Figure 19 shows as example of quantum mechanical representation of (*bra – ket*) vectors and calculation of quantum states as density matrices in SW.

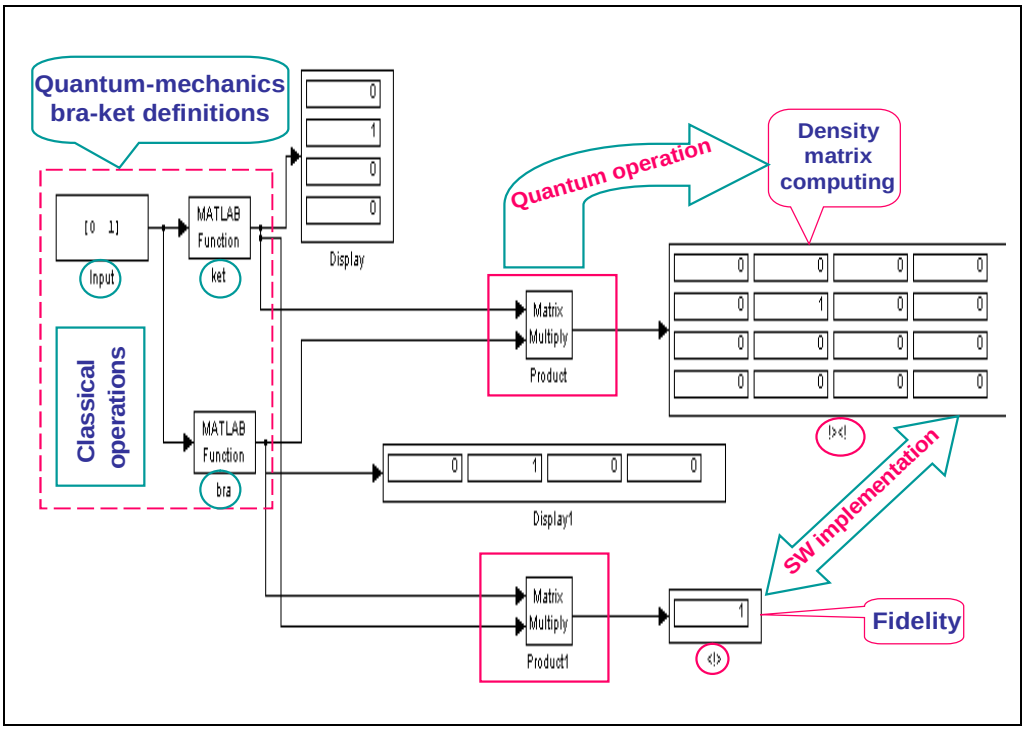

Figure 19: *SW representation of density matrix and fidelity calculation*

Example 2: *Quantum Shor's Algorithm* (*Quantum factorization promise*). Figures 11 and 13 shows that in quantum Shor algorithm U_F block that is a diagonal matrix of $2^{2n} \times 2^{2n}$ dimension. Finally output of entanglement is processed by interference block composed of *QFT* and identity matrix *I*. The output of entire algorithm is therefore the vector obtained after application of operator $QFT_n \otimes {}^n I$. Factorization time using matrix and vector approach are here reported (see, Figure 20). A standard PC (Pentium III 800 MHz 1Gb RAM) is used for simulations [9].

The superiority of second approach is in this case evident in Figure 20.

Remark. A quite more difficult task is to deal with interference. In fact, differently from Entanglement, vectors are not composed by elements having only two possible values. Moreover, the presence of tensor products, whose number increases dramatically with the dimensions, constitutes a critical point at this step. In order to find a suitable input-output relation, some particular properties of matrix $QFT_n \otimes {}^n I$ has been taken in consideration, where

$$[QFT_n]_{i,j} = \frac{1}{2^{n/2}} e^{2\pi J \frac{(i-1)(j-1)}{2^n}} .$$

The interference matrix $QFT_n \otimes {}^n I$ has several nonzero elements.

More exactly, it has $2^n \left(2^n - 1\right)$ zeros on each column. In order to avoid trivial products some modification can be made. If *Y* is the interference output vector, its elements y_i are

$$\mathrm{Re}[y_i] = \sum_{j=1}^{2^n} g_{(i \bmod 2^n)+2^n(j-1)+1} \cos\left(2\pi(j-1)\left(\mathrm{int}\left(\frac{i-1}{2^n}\right)\right)/2^n\right)$$

$$\mathrm{Im}[y_i] = \sum_{j=1}^{2^n} g_{(i \bmod 2^n)+2^n(j-1)+1} \sin\left(2\pi(j-1)\left(\mathrm{int}\left(\frac{i-1}{2^n}\right)\right)/2^n\right)$$

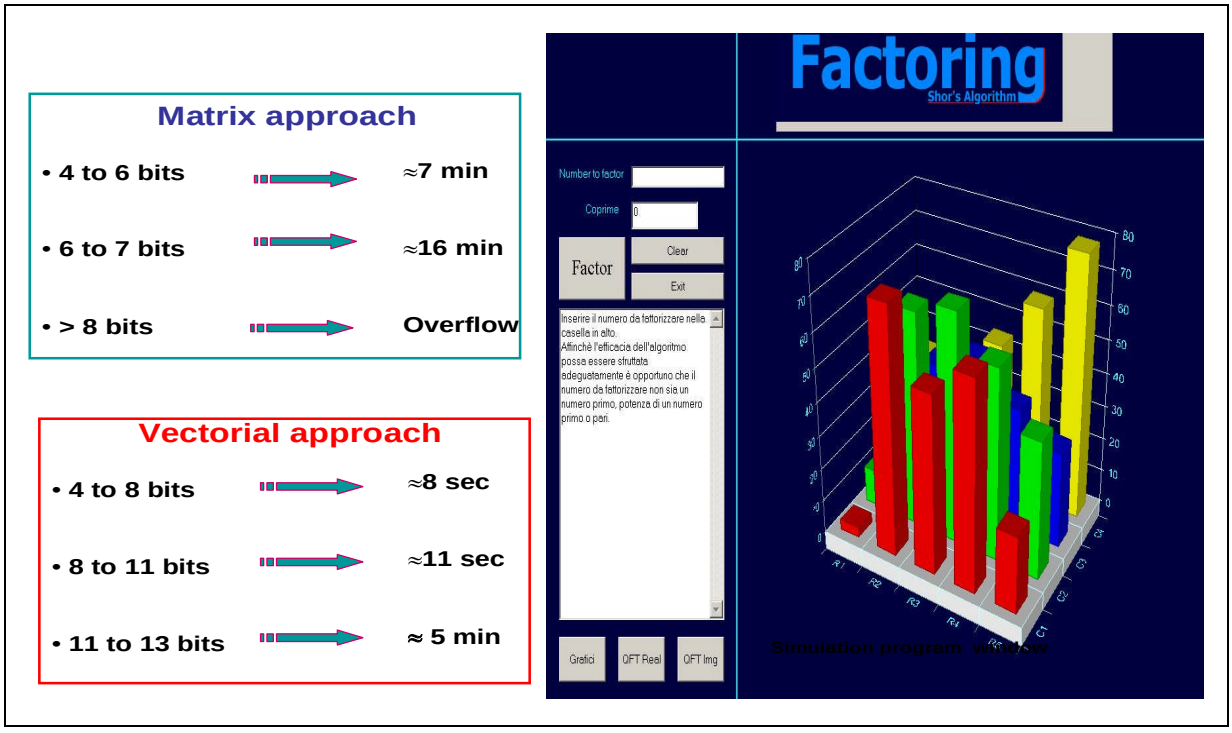

Figure 20: *SW simulation of Shor's quantum factorization algorithm*

The final output vector is therefore the following: $Y = \left[\mathrm{Re}(y_i) + j\,\mathrm{Im}(y_i)\right]$.

Example 3: *SW implementation of Grover's quantum search algorithm*. Figure 21 shows the simulation result of Grover's algorithm [problem-oriented approach with compressed vector allocation (see, Figure 2)].

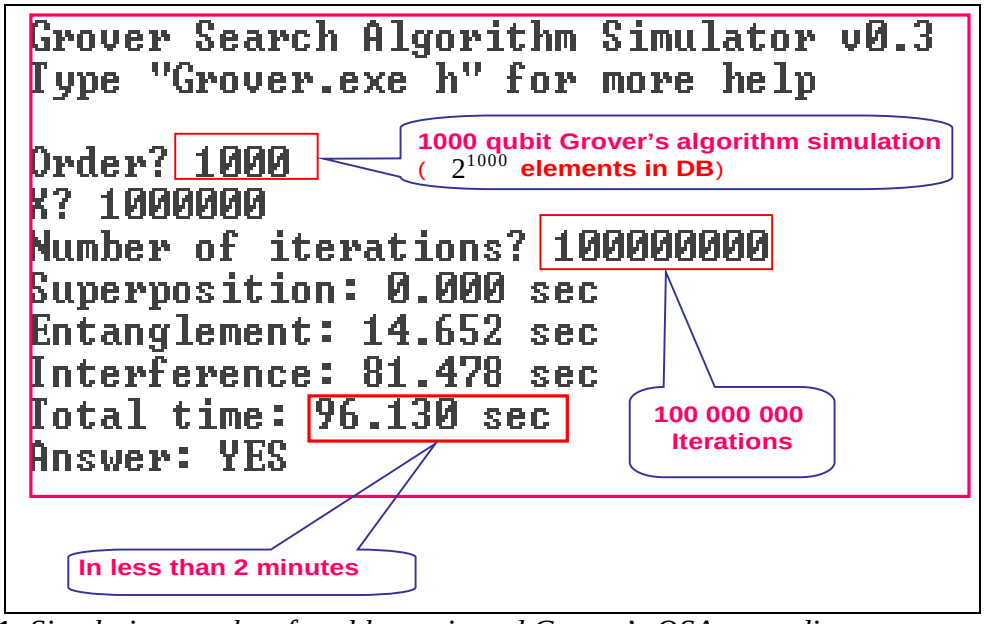

Figure 21: *Simulation results of problem oriented Grover's QSA according to approach 4 with 1000 qubit* (Simulator window snapshot)

Figure 22 shows optimal number of iterations for different qubit numbers and corresponding Shannon entropy behavior of Grover's quantum search algorithm [8].

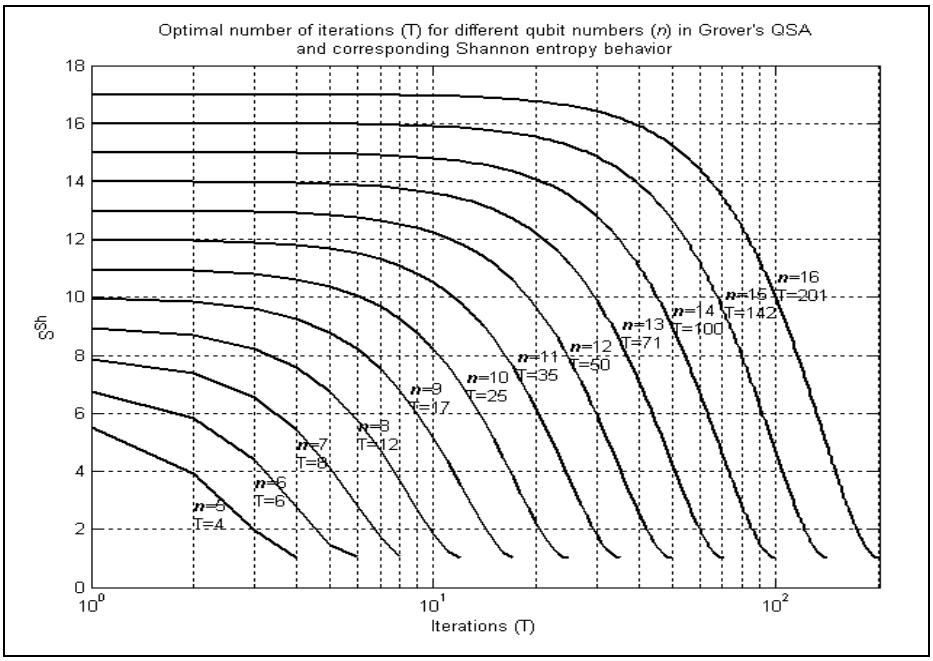

Figure 22: *Optimal number of iterations for different qubit numbers and corresponding Shannon entropy behavior of Grover quantum algorithm*

Let us briefly consider the QFMS application to design of intelligent control with quantum approach [10, 11].

4. Intelligent control of nonlinear dynamic system based on quantum search algorithm

Structure of intelligent control system is developed in [10]. Figure 23 shows the structure of nonlinear intelligent control system based on SCO and QCO.

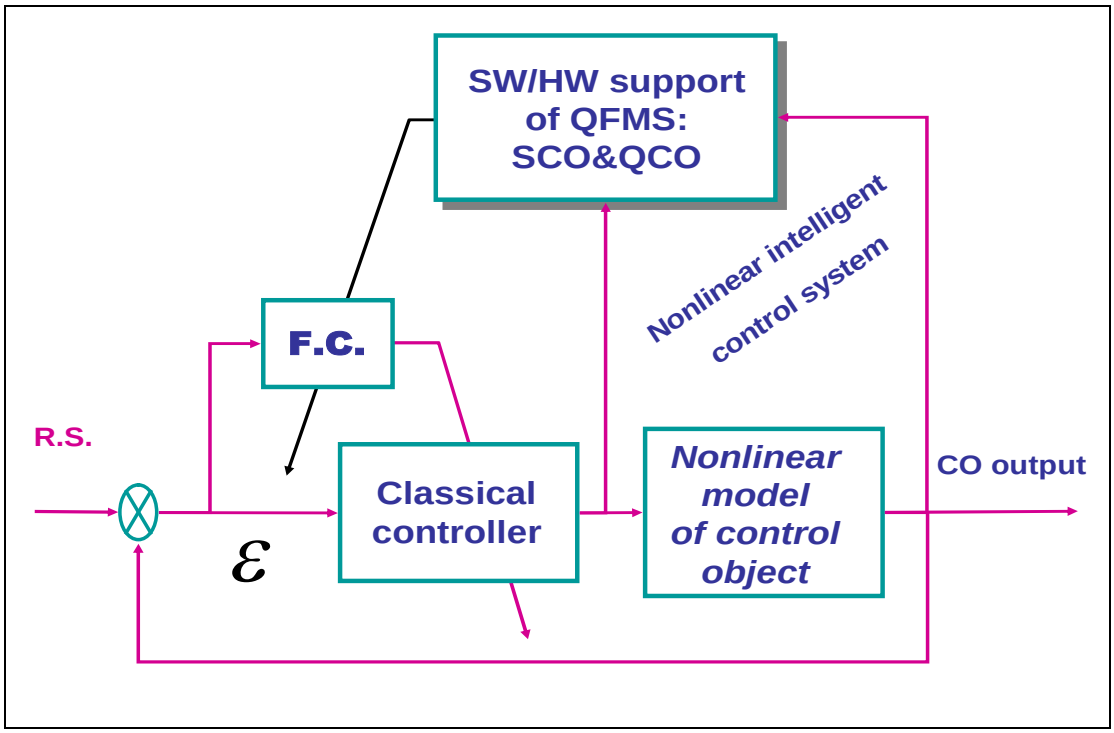

Figure 23: *Structure of intelligent control system based on* SCO&QCO *of KB*

Let us consider example of QFMS application in Figure 23 (as quantum control algorithm from Figure 1) based on Grover's quantum search algorithm (see, Figures 16 and 18).

Figure 24 shows the general structure of coefficient gain schedule design of fuzzy PID controller according to the structure of Grover's algorithm (see, Figure 16).

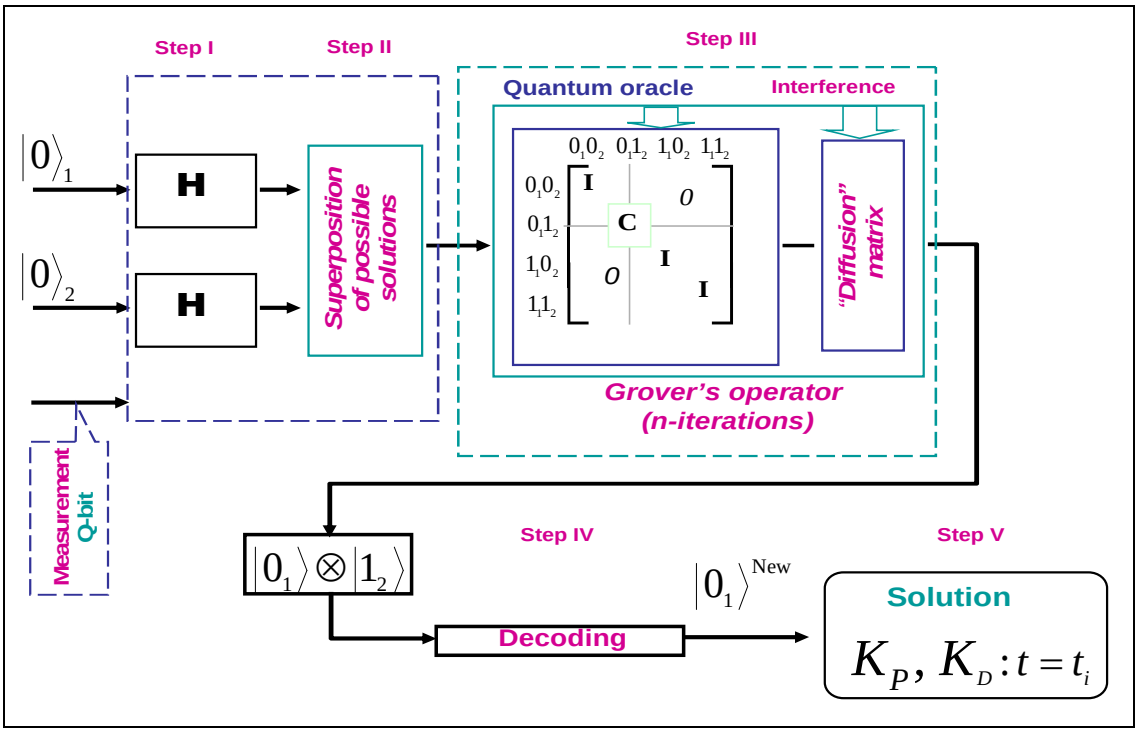

Figure 24: *Coefficient gain schedule design of fuzzy PID controller using Grover's algorithm*

Quantum superposition preparation with Hadamard transform H (Steps I and II) of two coefficient gain schedules (designed from SCO) is shown in Figure 25 and is used for design of a new coefficient gain schedule.

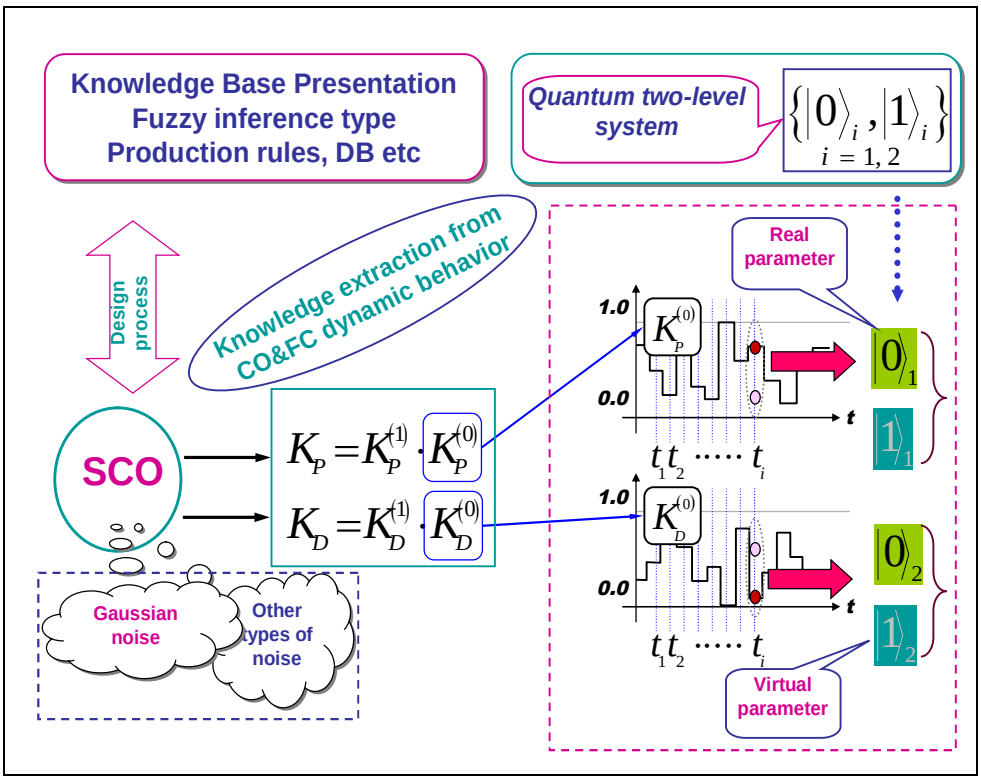

Figure 25: *Superposition of two classical coefficient schedules* (PD-controllers)

Figure 26 shows the design process of a new coefficient schedule of PD-controller.

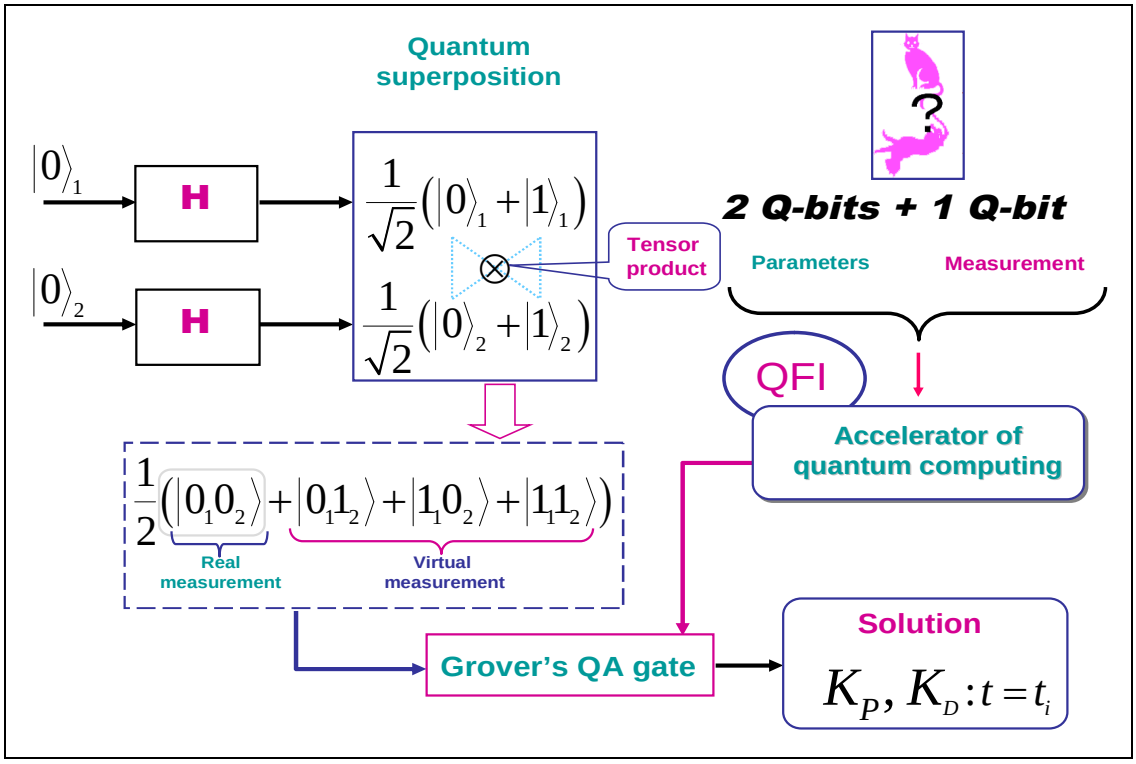

Figure 26: *Design of a new coefficient gain schedule of PD-controller*

Action of Hadamard transform $H=\frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix}$ on initial classical state $|0_1\rangle$ gives quantum superposition of two classical states:

$$H|0_1\rangle=\frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix}|0_1\rangle=\frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix}\begin{pmatrix}1\\ 0\end{pmatrix}=\frac{1}{\sqrt{2}}\begin{pmatrix}1\\ 1\end{pmatrix}=\frac{1}{\sqrt{2}}\left[\begin{pmatrix}1\\ 0\end{pmatrix}+\begin{pmatrix}0\\ 1\end{pmatrix}\right]=\frac{1}{\sqrt{2}}(|0_1\rangle+|1_1\rangle)$$

Tensor product application created full possible states of coefficient gain schedule and introduced quantum correlation between corresponding coefficients. Extraction of necessary information about searching solution is realized with quantum oracle.

Step III of design process in Figure 24 is realized with Grover's quantum search algorithm. Quantum oracle searches the best solution between two current values of coefficient gains of FC_g and FC_{ng} using corresponding position of negation $C=\begin{pmatrix}0 & 1\\ 1 & 0\end{pmatrix}$ (see, Figure 24) and decoding (step IV in Figure 24).

KB of first fuzzy controller (FC_g) is designed with SCO toolkit for Gaussian external noise; KB of second fuzzy controller (FC_{ng}) is designed with SCO toolkit for non-Gaussian external noise.

For the realization of Grover's QA gate accelerator of quantum computing is used (see, Figure 18). Computation control of quantum operations is realized according to the structure in Figure 5.

Figure 27 shows simulation results of CO dynamic behavior (as nonlinear Van der Pol oscillator) and coefficient gain schedule of PD-controller in a new unpredicted control situation (with new non-Gaussian external noise).

Simulink model and MatLab functions are used for stochastic simulation of dynamic behavior of control object and intelligent control.

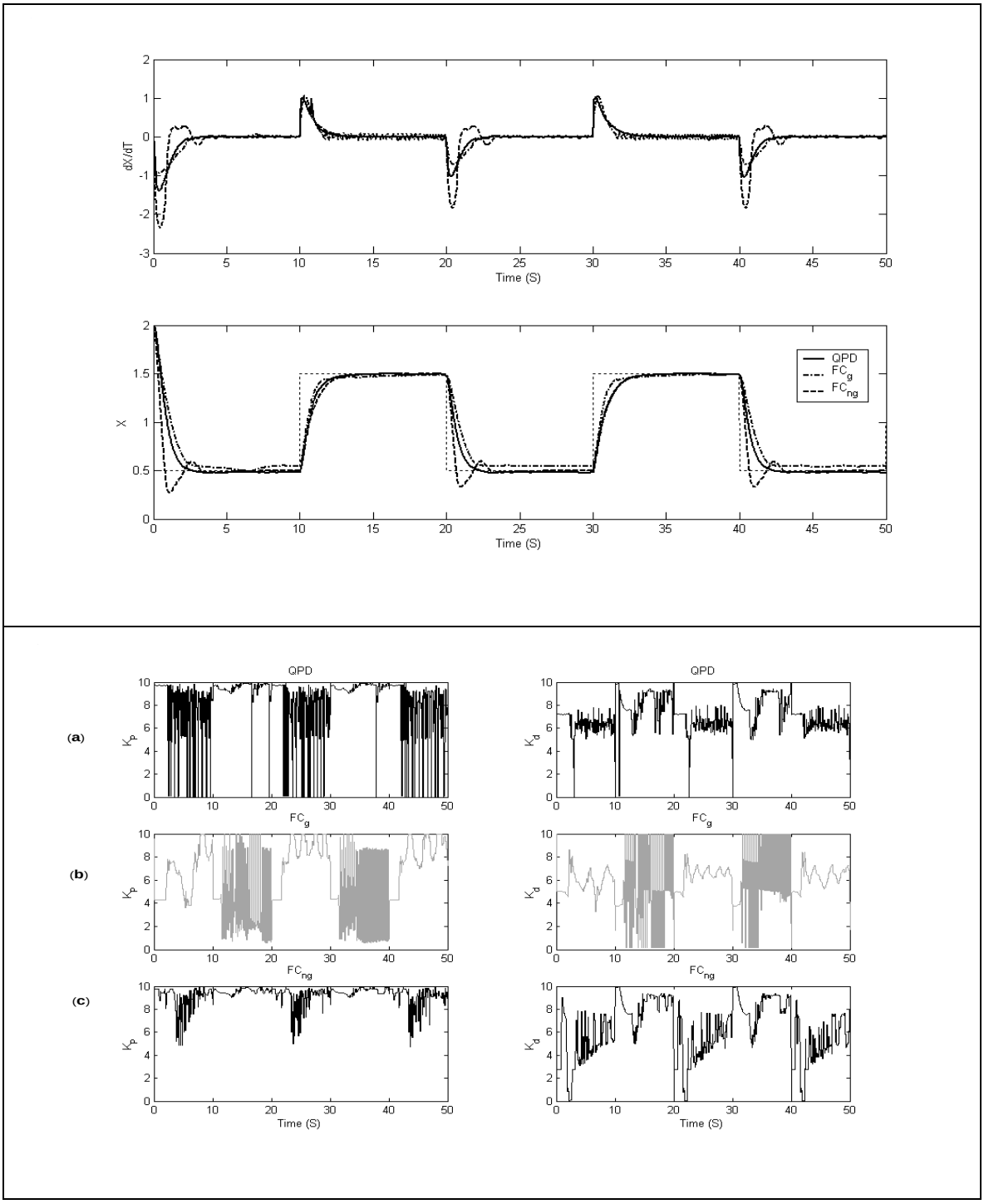

Figure 27: *Simulation results of dynamic behavior of control object (Van der Pol oscillator) and coefficient gain schedule of quantum PD-controller*

Results shows that for new unpredicted control situation it is possible achieve more accurate control with quantum search algorithm.

Conclusions

1. Flexible structure of Quantum Fuzzy Modeling System (QFMS) is developed.
2. SW/HW support of Quantum Modeling System (QMS) is described.
3. Realization on classical computer of quantum control algorithms with quantum search algorithm is introduced.
4. Application effectiveness of QFMS in design of intelligent fuzzy control is demonstrated.

References

1. **S.V. Ulyanov, L.V. Litvintseva, I.S. Ulyanov and S.S. Ulyanov, "***Quantum Control Algorithm of Self-Organization and Quantum Fuzzy Inference*: *Quantum Information* & *Computing Technology in Robust KB Design*," In this issue.
2. **M.A. Nielsen, I.L. Chuang,** *Quantum Computation and Quantum Information,* **Cambridge Univ. Press**, UK, 2000.
3. **S.V. Ulyanov, L.V. Litvintseva, I.S. Ulyanov and S.S. Ulyanov,** *Quantum information and quantum computational intelligence: Quantum decision making and search algorithms*, **Note del Polo Ricerca**, Università degli Studi di Milano (Polo Didattico e di Ricerca di Crema), Vols. 84 & 85, Milan, 2005.
4. **S.V. Ulyanov**, "*System and method for control using quantum soft computing*," **US patent** No 6,578,018B1, 2003.
5. **L.K. Grover**, "*A Fast Quantum Mechanical Algorithm for Database Search*," **US** *Patent* No 6,317,766 B1, 2001.
6. **Panfilov S.A., Litvintseva L.V., Ulyanov S.V.** *et all* "*Soft computing optimizer for intelligent control systems design: the structure and applications*," **J. Systemics, Cybernetics and Informatics (USA)**. 2003. Vol. 1, No 5.
7. **S.V. Ulyanov, G.G. Rizzotto, I. Kurawaki, S. Panfilov, P. Amato and D. Porto**, "*Method and hardware architecture for controlling a process or for processing data based on quantum soft computing*," **PCT** *Patent* WO 01/67186 A1, 2000.
8. **M. Branciforte, A. Calabrò, D. M. Porto, and I.S. Ulyanov**, "*Hardware design of main quantum algorithm operators and application in quantum search algorithm of unstructured large data bases*," **Proc. of the 7th World Multi-Conference on Systems, Cybernetics and Informatics** (***SCI*** '2003), Florida, Orlando July 27-30, USA, 2003.
9. **D.M. Porto, S.V. Ulyanov, K. Takahashi, S.A. Panfilov and I.S. Ulyanov**, "*Hardware implementation of fast quantum searching algorithms and its applications in quantum soft computing and intelligent control*", **Proc. Word Automation Congress** (WAC*'2004*), Seville, Spain, 2004.
10. **S.V. Ulyanov, K. Takahashi, L.V. Litvintseva** *et all* "*Quantum soft computing via robust control*: *Classical efficient simulation of wise quantum control in non-linear dynamic systems based on quantum game gates*," **Proc. of 2nd Internat. Conf. on Soft Computing, and Computing with Words in System Analysis, Decision and Control** (ICSCCW'*2003*), Antalya, Turkey, 2003. pp. 11 – 41.
11. **S.V. Ulyanov, K. Takahashi, I.S. Ulyanov, P. Amato, D.M. Porto and G.G. Rizzotto**, "*Quantum soft computing via robust control*: *From structure and HW-implementation of quantum algorithm gates to classical efficient simulation of robust control*," **Proc. ICAFS'*2005***, Antalya, Turkey, 2005.