SIMULATING AND EXECUTING CIRCUITS EMPLOYING THE QUANTUM COMPUTING PARADIGM

Daniel Abellán
Antonio Valdivia
Alberto A. Del Barrio
Guillermo Botella

Dept. of Computer Architecture and Automation
Universidad Complutense de Madrid
Prof. José García Santesmases 9,
Madrid, Spain
{dabellan,antvaldi,abarriog,gbotella}@ucm.es

Ginés Carrascal

IBM
Sta. Hortensia 26-28,
Madrid, Spain
gines_carrascal@es.ibm.com

ABSTRACT

Although Quantum Computing was introduced several decades ago, today is a hot-topic under research and development. Due to the technological difficulties to fabricate this type of computers, the employment of simulators become critical in order to allow researchers developing algorithms that can leverage the unique features of quantum computers, such as the case of Shor's or Grover's algorithms. Then, it is essential for these simulations to be as close as possible to the real results. In this paper we present a comparison between simulations and real executions of a 2-qubit adder. As the experiments show, the backends provided by IBM are able to obtain comparable simulated results to those obtained in real quantum computers.

Keywords: Quantum Computing, simulation, adder, QFT.

1 INTRODUCTION

At the beginning of the 80's the Nobel prize-winning physicist Richard Feynman postulated that to simulate quantum systems we would need to build quantum computers, a chimaera by this time. Nowadays many universities, major companies, startups have invested a vast amount of money on the promising topic of *Quantum Computing* (QC) (Cusumano 2018, Russell 2018) and even IBM has released the IBM Q System One: the first commercial quantum computer, which possesses 20 qubits (Berboucha 2019).

QC is based on three quantum-mechanical effects: superposition, entanglement, and quantum tunneling. (Coles et al. 2018). QC has the potential of drastically diminishing both execution time and energy consumption (Shor 1994, Shor 1997, Grover 1997) in comparison with conventional digital computing. Such is the case of the Shor's algorithm (Shor 1994, Shor 1997), which performs the factorization of an integer number in polynomial time, or the case of the Grover's algorithm (Grover 1997), which enables locating an element with a unique feature out of N elements in $O(N^{\frac{1}{2}})$ steps. These ideas together with the advancements on nano-manofacturing and the slow-down of CMOS-based hardware (Moore's Law, Dennard scaling

(Hennessy and Patterson 2012)) have raised the interest on QC significantly and posed the question on the arrival of the *Quantum Supremacy* (Calude 2019), which is estimated in 50-qubits.

Nevertheless, there is still a long path ahead, as there exist certain limitations due to technology, such as the depth or the coherence time (Sutor 2019), that avoid the deployment of large *quantum circuits*. It is therefore critical to simulate these quantum circuits under ideal and non-ideal conditions in order to prototype and verify suitable quantum algorithms. In this paper we present a comparison between quantum simulations with and without noise, and the results obtained on a real quantum computer. In order to perform these tests and illustrate several quantum effects, a 2-qubit adder has been employed as basic design.

The rest of the paper is organized as follows: Section 2 introduces basic concepts about QC, while Section 3 describes some works on quantum simulation and arithmetic units. Section 4 presents the basic circuit analyzed in this paper; Section 5 describes the simulation framework and Section 6 shows how these circuits simulate as well as the results of real executions. Finally, Section 7 draws our final conclusions.

2 QUANTUM PRELIMINARIES

In this Section, a brief overview about the basics on QC is given. The first concept to understand is the *qubit* (Coles et al. 2018). A qubit is a two-dimensional quantum-mechanical system that is in a state as pointed by Equation 1.

$$|\gamma\rangle = \alpha|0\rangle + \beta|1\rangle \ , \tag{1}$$

where $|0\rangle = \begin{pmatrix}1\\0\end{pmatrix}$ and $|1\rangle = \begin{pmatrix}0\\1\end{pmatrix}$ are abbreviations for representing the two basis states. In this definition, $\alpha, \beta \in \mathbb{C}$ and satisfy the condition $|\alpha|^2 + |\beta|^2 = 1$. Furthermore, if the qubit gets measured it will possess a classical bit value 0 with probability $|\alpha|^2$ or a classical bit value 1 with probability $|\beta|^2$.

An alternative representation of a qubit is given by the Bloch Sphere. As can be observed in Figure 1, any qubit is represented as a three-dimensional vector where the state $\alpha|0\rangle$ corresponds with the vector (0, 0, 1) while the state $\alpha|1\rangle$ corresponds with the vector (0, 0, -1). Whenever a superposition occurs, i.e. a and b are different to 0 in Equation 1, the vector will point somewhere between those two states in the sphere.

A system with various qubits is described by the tensor product $\otimes$. For example, in a system composed of three qubits where every qubit is in the state $|\gamma_j\rangle = \alpha_j|0\rangle + \beta_j|1\rangle$, being $j = 1,2,3$, the joint state is as described by Equation 2.

$$\begin{aligned} |\gamma_1\gamma_2\gamma_3\rangle &= |\gamma_1\rangle \otimes |\gamma_2\rangle \otimes |\gamma_3\rangle \\ &= \alpha_1\alpha_2\alpha_3|000\rangle + \alpha_1\alpha_2\beta_3|001\rangle + \alpha_1\beta_2\alpha_3|010\rangle + \alpha_1\beta_2\beta_3|011\rangle \\ &\quad + \beta_1\alpha_2\alpha_3|100\rangle + \beta_1\alpha_2\beta_3|101\rangle + \beta_1\beta_2\alpha_3|110\rangle + \beta_1\beta_2\beta_3|111\rangle \ . \end{aligned} \tag{2}$$

A measurement of all three qubits could result in any of the eight possibilities associated with the eight basis vectors. One can see from this example that the state space grows exponentially in the number of qubits n and that in general the number of basis vectors is 2^n.

As in the case of classical computing, there exist several gates responsible for performing operations among the qubits, where any operation must mathematically correspond with a unitary transform, in such a way that the operation is reversible. The most common gates are:

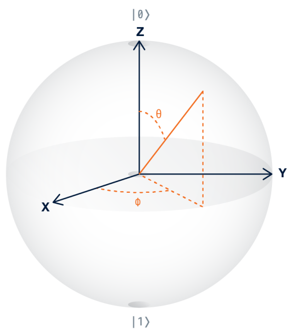


Figure 1: Bloch-sphere model (IBM 2019a).

- The *Pauli X* gate, which is the quantum version of the NOT gate. It performs a π radians rotation around the X axis.
- The *Pauli Z* gate, which performs a π radians rotation around the Z axis. As a variation of this gate, there exist several Z^k gates, $k = \frac{1}{2}, \frac{1}{4}, \frac{1}{8}$, in which the rotations around the Z axis are $\frac{\pi}{2}, \frac{\pi}{4}$ and $\frac{\pi}{8}$ radians, respectively.
- The *controlled NOT* (CNOT) which negates a qubit if the control qubit is $|1\rangle$.
- The *Toffoli* gate (T), aka *controlled-controlled NOT*, is the quantum version of the AND gate. It negates a qubit iff both control bits are $|1\rangle$.
- The *Hadamard* gate (H), which makes a π radians rotation around the X+Z axis. Basically it maps the X coordinate to Z and vice versa. This gate is necessary to perform superpositions.

2.1 QFT

The Fast Fourier Transform algorithm (Barrio et al. 2016a) can also be employed in a quantum computer. In fact, most quantum algorithms directly or indirectly employ the quantum version of Fourier transforms, as in the case of the Shor's algorithm (Shor 1994, Shor 1997).

Without going into much details, the Quantum Fourier Transform (QFT) (Coppersmith 1994, Barenco et al. 1996, Hirvensalo 2012) applies an H gate per qubit and a set of controlled rotations (Z^k). This scheme allows then transforming a binary value into a phase inside the Bloch sphere. Theoretically, in this way an n-bit value could be mapped on a qubit. However, this process would not be reversible, as the only *observable* states are $|0\rangle$ and $|1\rangle$. It is then necessary to employ n-qubits for encoding an n-bit value. In this manner, every encoded qubit possesses partial information of the entire number. The Least Significant Qubit (LSQ) has one bit encoded, the second two, and so on until the Most Significant Qubit (MSQ), which has the entire number encoded. In Figure 2 we show a 4-qubit QFT example. As can be observed, after the H gate is applied, A_3 goes through 3 controlled rotations, A_2 through 2 controlled rotations and so on.

Finally, it must be noted that besides the QFT, there is an Approximate QFT (AQFT) (Coppersmith 1994, Draper 2000). As the number of qubits grows, the number of rotations for the most significant bits also increases. Such is the case of A_3 in Figure 2, for example. After applying the H gate on A_3, the result will be rotated $Z^{\frac{1}{2}}$ if A_2 is $|1\rangle$, and this result will be rotated $Z^{\frac{1}{4}}$ if A_1 is $|1\rangle$, and this result will be rotated $Z^{\frac{1}{8}}$ if A_0 is $|1\rangle$. In order to avoid a large amount of rotations, the AQFT considers that if the number of qubits is moderately high, there will be not only a lot of Z^k rotations, but very tiny rotations too, whose effect will become negligible and thus could be discarded. This method is not completely exact, but reduces the complexity of the QFT.

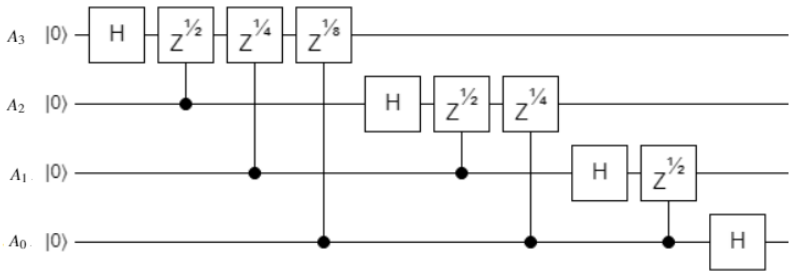


Figure 2: Circuit for a 4-qubit QFT.

3 RELATED WORK

As it has been mentioned in Section 1, QC has been achieving several important breakthroughs like the Shor's algorithms for prime factorization and the discrete logarithm problem (Shor 1994, Shor 1997). Nevertheless, despite the astonishing improvement, a part of the factorization algorithm must be performed in a classical computing. Hence, it would be desirable to deploy complete algorithms in the quantum computers. Therefore, as in the case of digital computing (Barrio et al. 2014, Barrio et al. 2016b, Kim et al. 2018a, Barrio et al. 2019), the employment of efficient arithmetic circuits is a must.

The first approaches consisted of following the classical structure of a ripple-carry adder (Vedral et al. 1997, Draper 2000, Cuccaro 2004, Wang et al. 2016). The work by (Vedral et al. 1997) employed blocks emulating the digital carry and sum calculation and required n extra ancillary bits to perform addition and possessed a depth in $O(n)$. In (Draper 2000) a scheme based on the QFT/AQFT is presented, requiring no ancillary bits and with a depth in $O(nlog(n))$. The work by (Cuccaro 2004) was based on the construction of a majority gate, requiring just one extra ancillary bit and with a depth in $O(n)$. In comparison with (Vedral et al. 1997), the overall complexity decreased, as the the amount of CNOT and T gates was modified from $4n+O(1)$ and $4n+O(1)$ to $5n+O(1)$ and $2n+1$, respectively. Finally, in the work presented by (Wang et al. 2016) there is a decrease in the number of T gates to just n, but at the expense of including n extra ancillary bits and increasing the depth to $3n+O(1)$.

As can be observed, comparing different designs is much more difficult than in the digital traditional scenario. Every design optimizes a parameter, and may be composed of different gates, as in the case of (Draper 2000), which uses H and Z^k gates while the others are composed of CNOT and T gates. There is no synthesis tool (Cong et al. 2011, Barrio et al. 2016a) and only few IBM quantum computers with few qubits are available to the public (IBM 2019b). Due to these reasons and other technological problems such as the coherence time, the calibration of the quantum computer, and others, simulating is still a must in QC (Sanchez-Palencia 2018, Calude 2019, Sutor 2019). Nevertheless, because of the aforementioned reasons, it must be highlighted that the simulated results are not usually the same under ideal conditions and under a real scenario. That is why in this paper we present a comparison of these scenarios to study such differences.

4 CASE STUDY

In this section we present a study on the quantum adder proposed by (Draper 2000). In comparison with other approaches based on classical structures as Ripple-Carry or Carry-Lookahead Adders (Vedral et al. 1997, Cuccaro 2004, Wang et al. 2016), this adder leverages the QFT. Basically it consists of three main blocks:

- QFT. One of the operands is transformed through the QFT. This input operand shall be named A, without loss of generality. This encoding enables the possibility of modifying the information just by using rotations in the Z axis. The output of this stage shall be named A^Q.
- Controlled rotations in the Z axis. In this block, the addition itself is performed by applying rotations on A^Q. The rotation is allowed or denied by the qubits of the second operator, i.e. B. If a qubit from B is $|1\rangle$ the rotation is done, and otherwise the input qubit from A^Q is forwarded. Each qubit of the QFT operator has partial information of the entire number, so partial rotations must be done to keep all the information safe. Given a qubit A_i^Q and a control qubit B_i, the rotation value is $\pi/2^{n-i+1}$, where n is the number of qubits. The total number of rotation gates can be easily computed then as $\sum_{j=1}^{n}\sum_{i=1}^{j} i \in O(n^2)$. Let's denote the outcome of this stage as $A^{Q'}$.
- QFT^{-1}. $A^{Q'}$ has the result of the addition but it is in superposition state, so the direct measurement would return a collection of all the possible additions, losing then the information of the operation. To avoid this, the reverse of the QFT must be done to extract the correct result. The complexity of this stage is exactly the same as in the QFT block.

An instance of the aforementioned adder for operands possessing 2-qubits is shown in Figure 3.

5 QUANTUM SIMULATIONS

The framework used for the experiments is Qiskit. Qiskit is an open-source quantum computing framework for leveraging today's quantum processors in research, education, and business (IBM 2019a, IBM 2019b). This framework provides a collection of simulation and real backends to test quantum circuits.

5.1 Unitary Simulator

This backend calculates the unitary matrix that represents the entire circuit. Every quantum gate has an unitary matrix that represents it. This backend multiplies the sequence of unitary matrix of every gate inside the circuit (not common matrix multiplication but Kronecker product). Given n qubits, the complexity of this operation $\in O(2^n)$. Getting the unitary matrix of significant functions (like addition, multiplication, division, etc.) can be an important advantage to simplify the circuits performing them to just one quantum gate. However, the rotations needed for applying this matrix maybe are arbitrarily small and impossible to perform in a real machine.

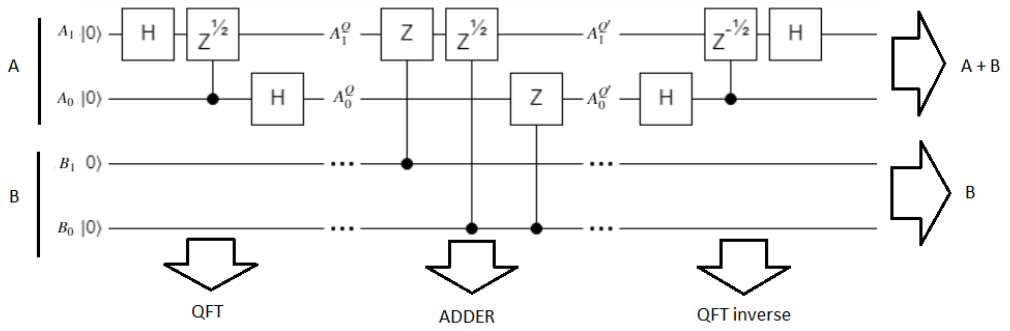


Figure 3: Circuit for a 2-qubit adder.

5.2 Circuit Simulations

The IBM quantum simulator executes a quantum circuit under certain conditions. Every execution shots an arbitrary number of times in different scenarios, namely: ideal conditions and noisy/real conditions.

5.2.1 Ideal Conditions

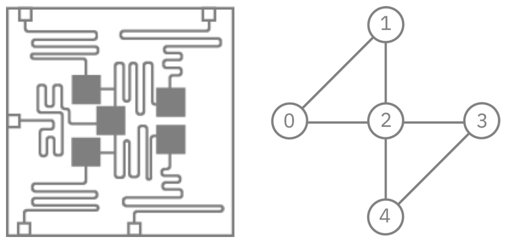
This type of simulations are performed by the *Statevector Simulator*, which is an auxiliary backend for Qiskit. The simulator just executes the circuit without any noise, unexpected delay, etc. and returns a histogram with the measurements.

5.2.2 Noisy Conditions

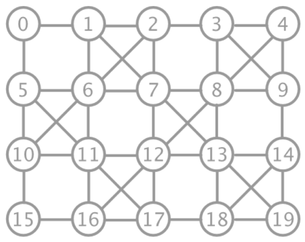
These simulations are carried out by the *QASM Simulator*. In order to do this, a real model of a quantum machine is provided to the simulator. The IBM framework provides real backends and the possibility to get a model of these real machines. Every real backend possesses a noise model, basis gates and a coupling map. The noise model defines the gate error factor of every gate, even the measurements, for each qubit (0 value in the ideal simulator). It is worthy to note that these models change every time the real machine is calibrated. Concretely, every time a backend is invoked through the simulator, the noise model is downloaded. The gate errors are in the order of 10^{-3} and in 10^{-2} for measuring. The basis gates are the gates implemented in the real machine. It must be noted that not all the gates are implemented, e.g. the T gate is not implemented. Concretely, the model of the real quantum machines that are used in Section 6 maps all the gates to a combination of gates named $U3$ and $U2$. Equation 3 describes the unitary matrix corresponding to $U2$.

$$U2 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & a_{11} & a_{12} \\ 0 & 0 & a_{21} & a_{22} \end{pmatrix} \quad , \quad a_{ij} \in \mathbb{C}, \forall i,j \ . \tag{3}$$

The coupling map is a graph that defines the interconnection between the qubits. In the ideal simulator the graph is complete, but in real conditions it is not, which means that not all the qubits can be connected through a gate. Figures 4a and 4b show the coupling maps for the 5-qubit *IBM Q Tenerife* and the *20-qubit IBM Q Tokio*, respectively. For instance, in the 5-qubit computer, qubits 1 and 3 cannot be connected directly.



(a) *IBM Q 5 Tenerife*.



(b) *IBM Q 20 Tokio*.

Figure 4: Coupling maps of IBM quantum computers (IBM 2019b).

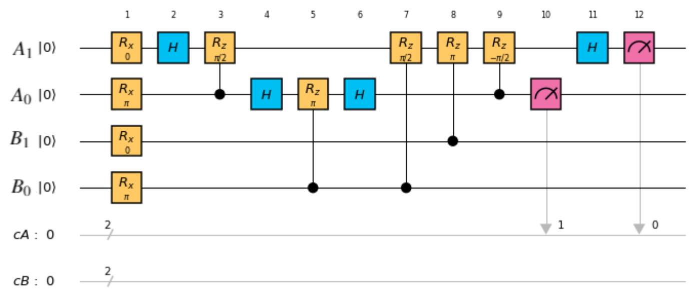


Figure 5: Modular adder circuit.

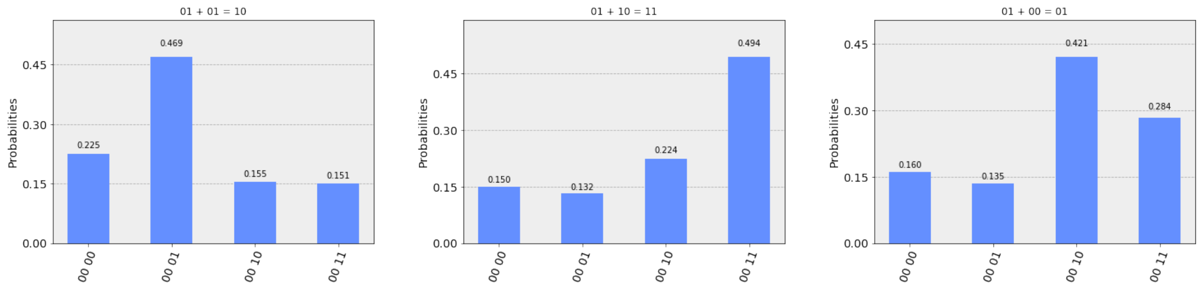
6 EXPERIMENTS

In this section we present our experiments. The simulations have been realized using the IBM Qiskit framework (IBM 2019a) running 10^3 times every test.

6.1 Simulation Results

In this subsection the performance of a 2-qubit modular adder will be evaluated. Figure 5 shows the design that will be evaluated in this first test. As can be observed, in comparison with the generic design depicted in Figure 3 it includes an initial rotation in the X axis for all qubits in order to initialize them. It must be noted that the initial value in a Qiskit simulation is always all 0's. Thus, initially the values to be added must be set through Pauli X gates. For instance, Figure 5 shows an initialization to $|01\rangle$ in both operands. When rotating 0 radians the qubit value will be set to $|0\rangle$ and when rotating π it will be set to $|1\rangle$. Besides, it must be noted that a final stage of measurement in $|A_0A_1\rangle$ is also included, which is where the results will be stored. In addition to this there are a couple of variables named *cA* and *cB* that represent the classical register on which the measurement is dumped.

Figure 6 shows the resulting histograms after simulating the 2-qubit adder in noisy conditions. In this case, we have employed the coupling map corresponding to the 5-qubit quantum computer *IBM Q 5 Tenerife*, which is the one depicted in Figure 4a. Every chart contained in this figure shows the possible qubit joint state in the X axis and the probability in the Y axis. Due to the way Qiskit arranges the inputs (the bottom-most is the most significant), in this case our joint state is $|B_0B_1A_0A_1\rangle$. Furthermore, it should be reminded that the result of the proposed adder is calculated on the first operand, i.e. A. Hence, Figure 6a should be interpreted as follows: the probability of the joint state to be $|0000\rangle$ is 0.225, to be $|0001\rangle$ is 0.469 and so on. Analogously with Figures 6b and 6c. On the contrary, under ideal conditions the result is always correct



(a) 01 + 01 (b) 01 + 10 (c) 01 + 00

Figure 6: 2-qubit additions running a noisy simulation.

with a probability of 1. Finally, it must be noted that $|B_0B_1\rangle$ is always $|00\rangle$ in these figures because it is never measured.

As can be observed in Figures 6a, 6b and 6c there appear several possible results for the same additions, with a different probability. That is the main difference with respect to a traditional computation circuit, although maybe QC has some similarities in this regard with how Convolutional Neural Networks detect an object (Kim et al. 2018b, Fabregat et al. 2017, Fariña et al. 2017). Although the correct results are still those obtaining the highest probabilities, they never surpass 50% probability, which is quite remarkable in comparison to the ideal simulation.

Another interesting effect to observe when simulating in non-ideal conditions is how the qubits interact among each other. For this purpose, the adder shown in Figure 7 has included an extra dummy bit D_0 that does not contribute to the circuit but affects the probability of getting the correct result, as observed in Figure 8a. Figure 8b illustrates the same additions but being run with the *IBM Q 14 Melbourne* backend model. In this case, the design shown in Figure 7 is completed with more dummy bits set to $|0\rangle$ that are not measured. As observed, the probabilities are also affected, although they remain close to those provided with the 5-qubit model in Figure 8a.

Furthermore, measuring the qubits has also an impact on such probabilities: measuring a qubit modifies the qubit itself (Lanzagorta and Uhlmann 2008, Coles et al. 2018). In order to test this, the circuit proposed in Figure 9 has been simulated. Although B_1, B_0 and D_0 do not interfere in the result calculation, they affect the final probability values, as can be observed in Figure 10. The result with the highest probability is still the correct one for the three cases observed in Figures 10a, 10b, 10c, but with a probability slightly greater than 0.3. It is quite remarkable to observe the explosion of probable results, in comparison with Figure 6 where only 4 results were possible.

6.2 Real Results

In this subsection we will show the execution results of the modular adders presented in Figures 5 and 9 in real backends provided by IBM. Figures 11a and 11b depict the results after executing the 2-qubit addition in the 5-qubit quantum computer *IBM Q 5 Tenerife* and the 14-qubits quantum computer *IBM Q 14 Melbourne*. For this test, the modular adder shown in Figure 7 has been considered (and extended for the case of the 14-qubit quantum computer). As it is observed, the results are similar than the simulated ones.

Finally, in Figure 12 the impact of measuring all the qubits is evaluated in the *IBM Q 5 Tenerife*. Once again, it is noteworthy to see how measuring dummy qubits influence the possible results. However, in spite of this and although the probabilities to get the correct results have decreased when comparing with Figure 10, the quantum computer is still giving the correct answer.

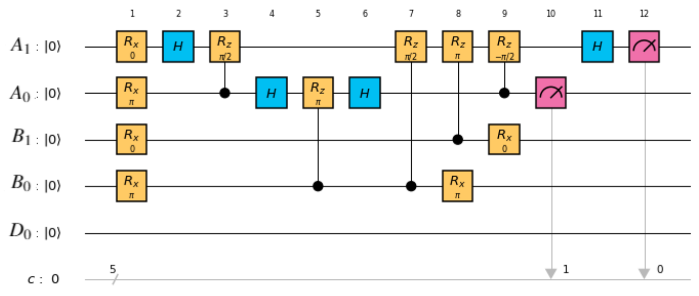
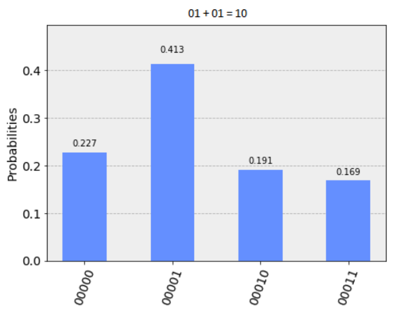
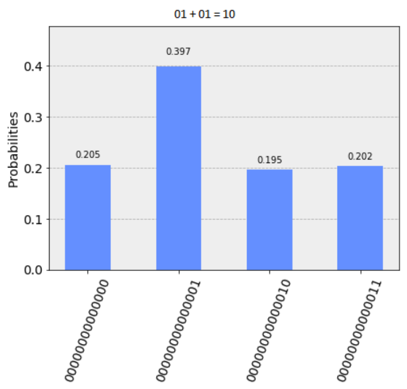


Figure 7: Modular adder circuit with dummy qubit.



(a) *IBM Q 5 Tenerife* model.



(b) *IBM Q 14 Melbourne* model.

Figure 8: 2-qubit additions running a noisy simulation without measuring dummy qubits.

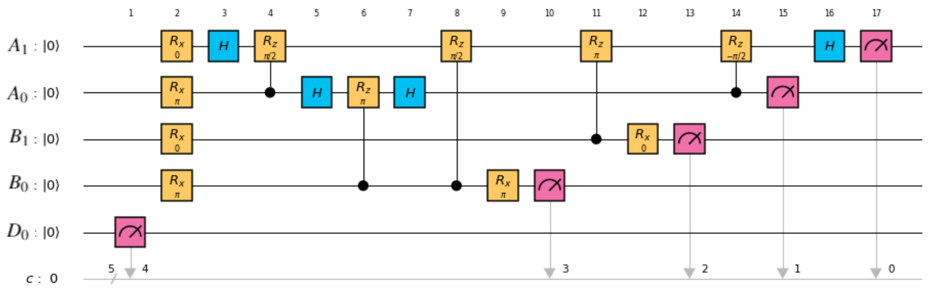


Figure 9: Modular adder with dummy qubit and measuring all the qubits

7 CONCLUSIONS

In this paper the basic features of QC have been studied by evaluating an adder based on the QFT. Simulations under ideal and noisy conditions have been performed and finally compared with executions on real backends provided by IBM. QC is still a topic to be researched and developed to scale towards larger designs, but simulators have proved to be close to the results provided by real backends, which encourage researchers in the development of novel algorithms able to make the most of this new paradigm of computation.

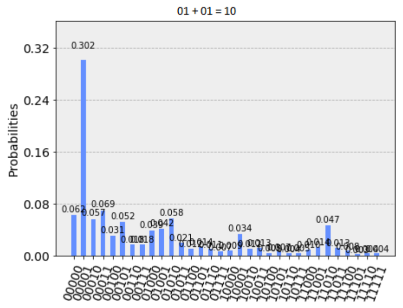
ACKNOWLEDGMENTS

This paper has been supported by the CM under grant S2018/TCS-4423, the EU (FEDER) and the Spanish MINECO under grant TIN 2015-65277-R and by Banco Santander under grant PR26/16-20B-1.

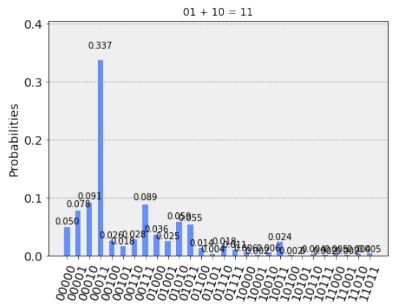
REFERENCES

Barenco, A. et al. 1996, july. "Approximate Quantum Fourier Transform and Decoherence". *Phys. Rev. Letters*, pp. 139–146.

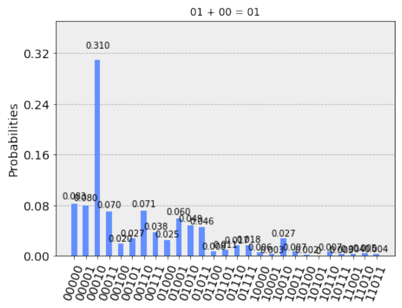
Barrio, A. A. D. et al. 2014, March. "Ultra-Low-Power Adder Stage Design for Exascale Floating Point Units". *ACM Trans. Embed. Comput. Syst.* vol. 13 (3s), pp. 105:1–105:24.



(a) 01 + 01

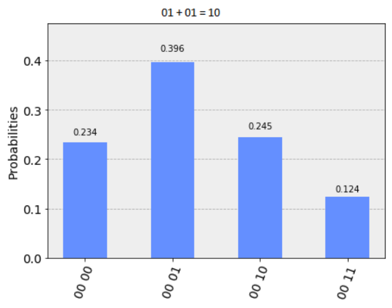


(b) 01 + 10

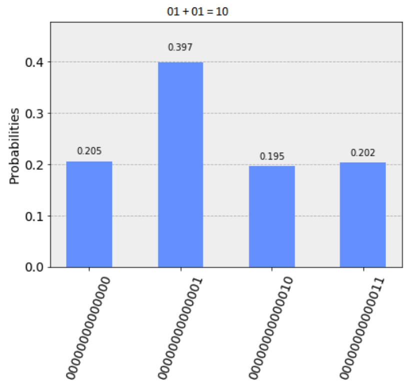


(c) 01 + 00

Figure 10: 2-qubit additions running a noisy simulation measuring dummy qubits.

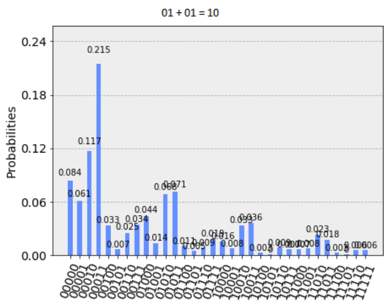


(a) *IBM Q 5 Tenerife*.

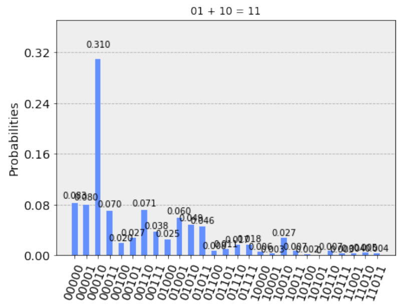


(b) *IBM Q 14 Melbourne*.

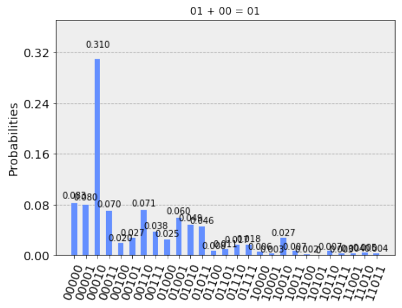
Figure 11: Results of executions on a real backends.



(a) 01 + 01



(b) 01 + 10



(c) 01 + 00

Figure 12: 2-qubit additions measuring dummy qubits after executing them on *IBM Q 5 Tenerife*.

Barrio, A. A. D. et al. 2016a. “A Distributed Clustered Architecture to Tackle Delay Variations in Datapath Synthesis”. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* vol. 35 (3), pp. 419–432.

Barrio, A. A. D. et al. 2016b. “A Partial Carry-Save On-the-Fly Correction Multispeculative Multiplier”. *IEEE Transactions on Computers* vol. 65 (11), pp. 3251–3264.

Barrio, A. A. D. et al. 2019. “A Combined Arithmetic-High-Level Synthesis Solution to Deploy Partial Carry-Save Radix-8 Booth Multipliers in Datapaths”. *IEEE Trans. on Circuits and Systems* vol. 66-I (2), pp. 742–755.

Berboucha, M. 2019. “IBM’s First Commercial Quantum Computer”. https://www.forbes.com/sites/meriameberboucha/2019/01/23/ibms-first-commercial-quantum-computer. [Online; accessed 25-February-2019].

Calude, C.S. et al. 2019. “The Road to Quantum Computational Supremacy”. https://arxiv.org/abs/1712.01356. [Online; accessed 25-February-2019].

Coles, P. J. et al. 2018. “Quantum Algorithm Implementations for Beginners”. *CoRR* vol. abs/1804.03719.

Cong, J. et al. 2011. “High-Level Synthesis for FPGAs: From Prototyping to Deployment”. *IEEE Trans. on CAD of Integrated Circuits and Systems* vol. 30 (4), pp. 473–491.

Coppersmith, D. 1994. “An approximate Fourier transform useful in quantum factoring”. https://arxiv.org/abs/quant-ph/0201067. [Online; accessed 25-February-2019].

Cuccaro, S.A. et al. 2004. “A new quantum ripple-carry addition circuit”. https://arxiv.org/abs/quant-ph/0410184. [Online; accessed 25-February-2019].

Cusumano, M. A. 2018. “The business of quantum computing”. *Commun. ACM* vol. 61 (10), pp. 20–22.

Draper, T.G. 2000. “Addition on a Quantum Computer”. https://arxiv.org/abs/quant-ph/0008033. [Online; accessed 25-February-2019].

Fabregat, H. et al. 2017. “Simulation and Implementation of a Low-cost Platform to Improve the Quality of Biological Images”. In *Proceedings of the Summer Simulation Multi-Conference, SummerSim*, pp. 25:1–25:9.

Fariña, D. et al. 2017. “A DCT and neural network based system to obtain the characteristics of biological images”. In *Proceedings of the Summer Simulation Multi-Conference, SummerSim*, pp. 26:1–26:11.

Grover, L. K. 1997. “Quantum mechanics helps in searching for a needle in a haystack”. *Phys. Rev. Letters*, pp. 325–328.

Hennessy, J. L., and D. A. Patterson. 2012. *Computer Architecture - A Quantitative Approach, 5th Edition*. Morgan Kaufmann.

Hirvensalo, M. 2012. “Mathematics for Quantum Information Processing”. In *Handbook of Natural Computing*, pp. 1381–1412.

IBM 2019a. “IBM Q Experience”. https://quantumexperience.ng.bluemix.net/qx/experience. [Online; accessed 25-February-2019].

IBM 2019b. “IBM Quantum devices & simulators - IBM Q”. https://www.research.ibm.com/ibm-q/technology/devices/. [Online; accessed 25-February-2019].

Kim, M. S. et al. 2018a. “Efficient Mitchell’s Approximate Log Multipliers for Convolutional Neural Networks”. *IEEE Transactions on Computers*.

Kim, M. S. et al. 2018b. “Low-power implementation of Mitchell’s approximate logarithmic multiplication for convolutional neural networks”. In *2018 23rd Asia and South Pacific Design Automation Conference (ASP-DAC)*, pp. 617–622.

Lanzagorta, M., and J. Uhlmann. (Eds.) 2008. *Quantum Computer Science*. Morgan & Claypool Publishers.

Russell, J. 2018. "Rigetti (and Others) Pursuit of Quantum Advantage". https://www.hpcwire.com/2018/09/11/rigetti-and-others-pursuit-of-quantum-advantage/. [Online; accessed 25-February-2019].

Sanchez-Palencia, L. 2018. "Quantum simulation: From basic principles to applications". https://arxiv.org/abs/1812.01110. [Online; accessed 25-February-2019].

Shor, P. W. 1994. "Algorithms for quantum computation: discrete logarithms and factoring". In *Proceedings 35th Annual Symposium on Foundations of Computer Science*, pp. 124–134.

Shor, P. W. 1997. "Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer". *SIAM J. Comput.* vol. 26 (5), pp. 1484–1509.

Sutor, B. 2019. "Gaining a Quantum Advantage". https://www.ibm.com/blogs/research/2018/10/quantum-advantage-2/. [Online; accessed 25-February-2019].

Vedral, V. et al. 1997, july. "Quantum networks for elementary arithmetic operations". *Phys. Rev. Letters*, pp. 147–153.

Wang, F. et al. 2016. "Improved quantum ripple-carry addition circuit". *Science China Information Sciences* vol. 59 (4), pp. 042406.

AUTHOR BIOGRAPHIES

DANIEL ABELLÁN is finishing the degree in Computer Science from the Complutense University of Madrid (UCM), Madrid, Spain. Since 2018, he has been researching Quantum Computing for his Graduation Project. His email address is dabellan@ucm.es.

ANTONIO VALDIVIA is studying both Mathematics and Computer Sciece in Complutense University of Madrid (UCM). He has received a grant for researching in Quantum Computing for the academic year 2018/2019. His email address is antvaldi@ucm.es.

ALBERTO A. DEL BARRIO received the Ph.D. degree in Computer Science from the Complutense University of Madrid (UCM), Madrid, Spain, in 2011. Since 2017, he has been an Interim Associate Professor of Computer Science with the Department of Computer Architecture and System Engineering, UCM. His research interests include Design Automation, Arithmetic as well as Video Coding Optimizations. His email address is abarriog@ucm.es.

GUILLERMO BOTELLA received the M.A. Sc. degree in Physics in 1998, the M.A.Sc. degree in Electronic Engineering in 2001 and the Ph.D. degree in 2007, all from the University of Granada, Spain. Currently he is an Associate Professor at the Department of Computer Architecture and Automation of Complutense University of Madrid, Spain. His current research interests include Digital Signal Processing for VLSI, FPGAs, GPUs, HPC and Vision Algorithms. His email address is gbotella@ucm.es.

GINÉS CARRASCAL received the M.A. Sc. degree in Physics in 1999, from the University of Salamanca, Spain. After post graduate studies joined IBM in 2000 as IT Architect, and currently working as Quantum Ambassador. Since 2017, he has been an Interim Assistant Professor of Computer Science with the Department of Software Systems and Computation, UCM. His research interests include applied artificial intelligence and Quantum Computing. His email address is gines_carrascal@es.ibm.com.