IMPLEMENTING QUANTUM ALGORITHMS IN IBM QUANTUM EXPERIENCE

Jose David Perdomo, Ingeniero Mecatrónico, Darwin Luque, Ingeniero Mecatrónico, Jose Luis Ordoñez Avila, Master en Administración de Proyectos, and Ivan de Jesus Deras, Master en Computación
Facultad de ingeniería, Universidad Tecnológica Centroamericana, Unitec, San Pedro Sula, Honduras,
jlordonez@unitec.edu, *ivan.deras@unitec.edu*

Abstract: ***Quantum computing has been researched and developed since 1960, even though it's still unknown to most people. Classic computer systems are reaching the physical limits of the construction size of transistors, a key device for their operation. Manufacturers are forced to reduce the size of these devices to include a bigger amount of transistors in their chips and thus achieve higher processing speeds. Up to date, there isn't a viable solution to this problem so the stakes are on quantum computing. This technology takes advantage of the properties of quantum physics to enhance its processing capabilities, such as superposition, entanglement, and even quantum tunneling, which is responsible for causing classic chips to fail at very small scales. As a technology under development, it still has room for improvement in error correction and processing speeds. The most prominent quantum algorithms were tested on the IBM Quantum Experience platform. The results obtained in these implementations were satisfactory, running the algorithms on a real quantum processor. Similarly, this document compares the IBM quantum processors and analyzes the differences in results obtained on the same algorithms based on the calibrations for each processor. It has been proven that each processor will obtain a higher success rate depending on the type of gate used, and the error rate for each gate. This error rates can be obtained from the processors calibration sheets. The effects of noise and decoherence are present, negatively affecting the results obtained specifically in larger algorithms.***

Keywords—IBM, quantum processors, quantum noise

Digital Object Identifier (DOI):
http://dx.doi.org/10.18687/LACCEI2021.1.1.451
ISBN: 978-958-52071-8-9 ISSN: 2414-6390

IMPLEMENTACIÓN DE ALGORITMOS CUÁNTICOS EN IBM QUANTUM EXPERIENCE

Jose David Perdomo, Ingeniero Mecatrónico, Darwin Luque, Ingeniero Mecatrónico, Jose Luis Ordoñez Avila, Master en Administración de Proyectos, and Ivan de Jesus Deras, Master en Computación
Facultad de ingeniería, Universidad Tecnológica Centroamericana, Unitec, San Pedro Sula, Honduras,
jlordonez@unitec.edu, ivan.deras@unitec.edu

Abstract*– La computación cuántica ha sido investigada y desarrollada desde 1960 aunque aún es desconocida para muchas personas. Los sistemas de computación clásicos están llegando a los límites físicos de la construcción de transistores, dispositivo clave para el funcionamiento de los mismos. Los fabricantes se ven obligados a reducir el tamaño de estos dispositivos para lograr incluir más en sus chips y así alcanzar potencias mayores. No parece existir una solución viable para esta problemática por lo que las apuestas están en la computación cuántica. Esta tecnología aprovecha los fenómenos de la física cuántica para potenciar sus capacidades de procesamiento, como ser la superposición, entrelazamiento e incluso el túnel cuántico, que es responsable de hacer fallar los chips clásicos a escalas muy pequeñas. Al ser una tecnología en desarrollo, aún tiene espacio de mejora en corrección de errores y velocidad de procesamiento. Se comprobaron los algoritmos cuánticos más prominentes en la plataforma cuántica IBM Quantum Experience. Los resultados obtenidos en estas implementaciones fueron satisfactorios, pudiendo comprobar con éxito el funcionamiento de estos algoritmos. De igual manera el presente documento hace una comparación entre los diferentes procesadores cuánticos de IBM y analiza las diferencias en resultados de los mismos algoritmos basándose en las calibraciones para cada procesador. Se ha logrado identificar que cada procesador obtendrá una tasa de éxito mayor dependiendo del tipo de compuerta utilizada, y la tasa de error para la misma obtenida de su hoja de calibración. Los efectos del ruido y la decoherencia son notables, afectando negativamente algunos resultados obtenidos específicamente en algoritmos de mayor tamaño.

Keywords—IBM, quantum processors, quantum noise

I. INTRODUCCIÓN

Las computadoras son por mucho los aliados más importantes para el sostenimiento de la civilización actual. Han pasado 150 años desde que apareció el primer prototipo de computadora, y estos solo se han vuelto mucho más potentes con el pasar del tiempo. En 1965, el cofundador de Intel, Gordon Moore, afirmaba que el número de transistores en un circuito integrado se duplicaría cada pocos años, y esta tendencia continuaría durante las próximas décadas. La duplicación de los transistores tuvo un impacto directo en el rendimiento, haciendo que los chips fueran más potentes y capaces de realizar tareas delicadas y desafiantes en tiempos de ejecución más bajos [1]. Los fabricantes actualmente están alcanzando el límite físico de los transistores, lo que hace que los chips integrados experimenten fallas debido al fenómeno del túnel cuántico. La computación cuántica no usa bits (0 y 1) como los sistemas clásicos, en cambio utiliza Qubits basados en un solo átomo. Los Qubits pueden tener valores 0 y 1 al mismo tiempo a través de un proceso llamado superposición cuántica. En contraste, el entrelazamiento cuántico dicta que es simplemente imposible definir a un Qubit que interactuó con otro con un valor propio [2]. El desarrollo de la tecnología cuántica se ha convertido en la nueva "carrera espacial" tecnológica. Muchos grandes fabricantes, como Google e IBM, han invertido continuamente una cantidad considerable de recursos para lograr la supremacía cuántica [3].

Hasta la fecha, no existe un método eficiente para simular un entorno cuántico en uno clásico. En la práctica, es imposible simular siquiera un circuito compuesto por una modesta cantidad de 50 Qubits [4]. Hasta la fecha, no existe un sistema cuántico completamente escalable. Sin embargo, los investigadores han encontrado muchas propiedades prometedoras en la computación cuántica. El entrelazamiento cuántico se propone como el futuro de la criptografía. Esto se logra mediante las propiedades de la mecánica cuántica. Cualquier intento de acceder a la información hará que el sistema colapse y pueda ser detectado rápidamente. Muchas operaciones que la computación clásica considera intratables pueden lograrse mediante un enfoque cuántico. El algoritmo de factorización de Peter Shor y el algoritmo de búsqueda de Lov Grover han demostrado ser más rápidos en la realización de tareas difíciles como la factorización y búsqueda de datos en comparación con la computación clásica [6]. Además, el algoritmo cuántico de Deutsch-Josza ha demostrado obtener una ventaja real explotando los principios de superposición y realizar una operación en la mitad del tiempo que una computadora clásica.

II. IBM QUANTUM EXPERIENCE

En mayo de 2016, IBM lanzó una campaña llamada IBM Quantum Experience o IBMQX por sus siglas en ingles. IBMQX es un procesador / simulador cuántico basado en la nube, que ejecuta algoritmos cuánticos [7]. IBMQX permite a los usuarios crear sus circuitos cuánticos y simularlos o ejecutarlos en tiempo real en un procesador cuántico n-Qubit. IBM Quantum Experience brinda la posibilidad de interactuar con procesadores cuánticos de hasta 15 Qubits y un simulador cuántico capaz de procesar hasta 32 Qubits.

Además, IBMQX proporciona un archivo de calibración para cada procesador cuántico, que describe las tasas de error en las puertas U de un solo Qubit y las puertas CNOT.

La Tabla 1 muestra todos los procesadores cuánticos disponibles proporcionados por IBM. Esta interfaz ejecuta un circuito cuántico deseado y proporciona información útil sobre la ejecución, como tiempos de ejecución y resultados probabilísticos. La ejecución de un circuito cuántico en un procesador cuántico real le da al usuario una idea del comportamiento real de los Qubits cuando se exponen al ruido y la decoherencia.

Tabla 1. Procesadores de IBMQX.

Nombre del procesador	Qubits
Simulator	32
Melbourne	15
Rome	5
London	5
Burlington	5
Essex	5
Ourense	5
Vigo	5
Yorktown	5
Armonk	1

El editor de circuitos proporciona todas las compuertas cuánticas universales útiles, que serán suficientes para construir cualquier circuito cuántico posible.

III. Algoritmo de Deutsch-Jozsa

El algoritmo de Deutsch-Jozsa (DJ) fue propuesto en 1992 por David Deutsch y Richard Jozsa. Este algoritmo tiene como objetivo resolver una función $f: \{0,1\} \rightarrow \{0,1\}$ que tiene como entrada una cadena de ceros y unos. Este algoritmo determina si una función f (x) está balanceado y genera 1 y 0 el 50% del tiempo o es constante y genera 1 o 0 siempre. Normalmente, un enfoque clásico para esta situación requeriría $2^{n-1}+1$ operaciones, pero los algoritmos de DJ pueden resolver esto en una sola iteración debido a la superposición cuántica [8]

El algoritmo de DJ se divide en tres etapas específicas, la preparación de la superposición, la función de oráculo y la medida. En resumen, el algoritmo DJ da como resultado uno si la función está equilibrada y cero si es constante [9]. La función para definir este comportamiento esta dada por la ecuación 1.

$$-1^{f(x)}|x\rangle \times \frac{|0\rangle - |1\rangle}{\sqrt{2}} \quad (1)$$

En el que x son n Qubits de entrada en superposición. La superposición de los Qubits de entrada es lograda mediante la aplicación de una serie de compuertas Hadamard. Al mismo tiempo, una función oráculo específica cambiará la entrada de acuerdo con su tipo (constante o balanceada). La medición final determinará la propiedad del oráculo en un solo paso, logrado a través de las maravillas de la superposición.

IV. Algoritmo de Grover

En 1996, Lov Grover propuso un algoritmo cuántico para realizar una búsqueda rápida en una base de datos. Grover afirmo que tal algoritmo podría reducir el tiempo de ejecución de O (N) a O($\sqrt{N}$)[10]. El algoritmo de Grover proporciona un aumento de velocidad en los tiempos de procesamiento de búsqueda de una base de datos sin clasificar. La computación cuántica podría proporcionar una ventaja de velocidad significativa sobre las computadoras clásicas. El algoritmo de Grover se basa en dos etapas. La primera es una inversión de fase en la que el signo de los números deseados cambia. La segunda etapa consiste en una inversión sobre la media, que potencia la separación de fases entre el número deseado y el resto para resaltar aún más la respuesta correcta [11]. El algoritmo de Grover es una aplicación específica del proceso de amplificación de amplitud [12]. Puesto de una manera más sencilla, se podría considerar el algoritmo de Grover como una combinación del algoritmo de Deutsch-Josza (inversión de fase) y amplificación de amplitud. La etapa de inversión de fase se puede definir en la ecuación 2.

$$\varphi = \begin{cases} -1|x\rangle \times \left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right] & if\ x = u \\ 1|x\rangle \times \left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right] & if\ x \neq u \end{cases} \quad (2)$$

Esto invierte el signo si el registro es la respuesta correcta. Posteriormente, se aplica la inversión sobre la media como se muestra en la ecuación 3. En el que v es el registro y a es el promedio. Esta inversión sobre la media combinada con la inversión de fase hace que la respuesta correcta resalte por sobre el resto.

$$v' = -v + 2a \quad (3)$$

El algoritmo de búsqueda del algoritmo de Lov Grover ha sido ampliamente discutido y se han propuesto mejoras para hacer este algoritmo más rápido y estable. Algunas de las mejoras propuestas se pueden encontrar en [13], que propone un método diferente sobre índices ponderados y emparejamiento de fases. Además, existen otras aplicaciones potenciales para este algoritmo, como la criptografía [14].

V. Algoritmo de Shor´s

La factorización de enteros ha sido uno de los mayores desafíos que ha tenido la computación moderna. Es tan inaccesible para las computadoras clásicas que la criptografía moderna se basa en la factorización de enteros y la exponenciación modular [15]. En 1994, Peter Shor propuso un algoritmo capaz de factorizar grandes números en una

computadora cuántica, lo que atrajo una cantidad significativa de atención hacia el entorno cuántico [16]. El algoritmo de Shor es uno de los algoritmos más destacados en la computación cuántica y se ha propuesto como una amenaza real para los códigos criptográficos como el sistema de clave pública RSA [17]. El algoritmo de Peter Shor es capaz de factorizar enteros en tiempo polinomial y funciona bajo el hecho de que la factorización de un número puede reducirse a encontrar un período de una función particular denotada por $f(x) = a^x mod\ N$ [11]. Este algoritmo fue comparado con el algoritmo clásico más conocido, el General Number Field Sieve, concluyendo que el algoritmo cuántico es más rápido factorizando enteros de más de 1000 decimales [18].

El algoritmo de Shor se puede dividir en tres etapas principales, la superposición de entrada inicial, una función de exponenciación modular y una transformada cuántica de Fourier inversa para encontrar el período de la función. Dados n y x, el algoritmo de Shor tiene como objetivo encontrar el mínimo r tal que $x^r \equiv 1(mod\ n)[19]$. Una vez se ha encontrado el valor de r, los factores n pueden ser encontrados mediante el máximo común divisor descrito como $gcd(a^{\frac{r}{2}} \pm 1, C)$.

VI. Circuitos cuántica en FPGA

Es posible emular hasta cierto punto un sistema cuántico en un dispositivo FPGA, pues los arreglos de operadores lógicos permiten realizar más de una operación en paralelo simulando así la superposición. Bajo esta premisa, es necesario modelar un Qubit y las compuertas cuánticas basándose exclusivamente en las compuertas y bits clásicos. Por lo que en esta sección probaremos el algoritmo de Deutsch-Jozsa en una FPGA.

Se define que el estado de un Qubit en superposición esta dado por la ecuación:

$$|\alpha|^2 + |\beta|^2 = 1$$

Al ser un vector unitario, es posible definir la cantidad de bits necesarios para representar α y β para n cantidad de Qubits de la siguiente manera:

$$\begin{cases} a = 7 * 2^n \\ b = 1 * 2^n \end{cases}$$

Por consiguiente, se determina que la cantidad de bits necesarios para simular n cantidad de Qubits puede ser modelada por la ecuación

$$m = 2^n(7 + 2^n), \qquad n > 0$$

Si se hace una regresión a [4] en donde se afirma que es imposible simular más de 50 Qubits y se aplica la ecuación anterior, se obtiene que se requieren $4\ x\ 10^{100}$ bits en paralelo para procesar esta información.

Si bien el presente documento hace referencias a la emulación de circuitos cuánticos en un FPGA, se limitará a denotar las generalidades de funcionamiento y no su desarrollo a profundidad. La figura 1 muestra el circuito equivalente para ejecutar el algoritmo de Deutsch-Jozsa en un FPGA.

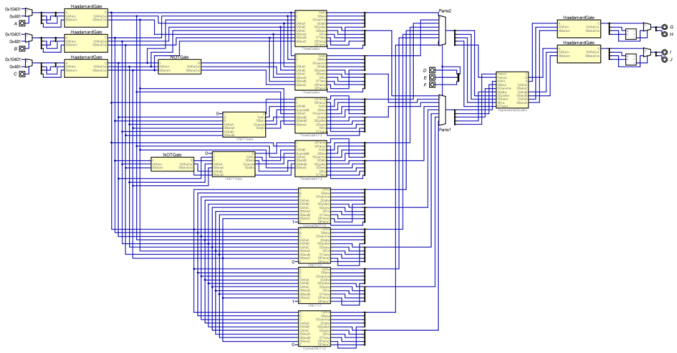


Figura 1. Circuito en FPGA para algoritmo de Deutsch-Jozsa

Como es apreciable, si bien es posible emular un circuito cuántico en un FPGA, se requiere el modelado de compuertas y Qubits lo que aumenta la dificultad proporcionalmente al aumentar la cantidad de Qubits. Se han ejecutado numerosos algoritmos en estos dispositivos con éxito, sin embargo tienen limitantes en la cantidad de Qubits a utilizar y su implementación posee una dificultad elevada.

En este caso para la generación del código no le se le realizará ninguna modificación al circuito ya que se diseñaron de manera que este ya esté en forma para medir su funcionalidad, y en caso de que funcione medir el rendimiento que tiene el FPGA para la emulación de estos algoritmos cuánticos. De tal forma que los circuitos creados en Digital se estarán exportando a Verilog por medio de la función disponible en el mismo software. Una vez exportado se procede a crear un proyecto en el software ISE Design Suite y abrir el código en Verilog en el ISE Design Suite. Abierto el código se corre la función Implement Top Module para verificar la implementación, sintaxis y síntesis del código generado. Obtenido que el código no tiene ningún problema se procedes a distribuirle los pines necesarios a través del archivo UCF.

Teniendo en cuenta que se tienen seis entradas y cuatro salidas se procede a distribuir las mismas en los recursos disponible en la placa Mimas V2. Como siempre seccionando las entradas al dip switch y las salidas a las luces LEDs. De tal manera entonces que el archivo UCF se ve de la siguiente manera.

Para comprender el funcionamiento se tiene que mantener en cuenta las posibles combinaciones del dip switch para los posibles casos para los tipos de funciones. Las funciones se controlan con los switches 6, 7 y 8 del dip switch por lo que su valor numérico es que tan significativo son. Por lo que cuando los tres switches están apagado o solamente el switch 6 esta encendido la función es constante. Cualquier caso distinto representa una función balanceada. Ahora también se tiene que mantener claro que representa cada LED. Las LEDs sobre la Mimas V2 se cuentan de izquierda a derecha. Por lo que el D1 y D2 es asignado al estado de respuesta del bit cuántico

más significativo, y D3 y D4 es asignado al estado de respuesta del bit cuántico menos significativo. Todo esto basado en los circuitos diseñados y en la distribución de las variables en los pines de la FPGA, considerando claro la posición de los componentes en la Mimas V2. Por lo que se nota en la ilustración 80 es que en ambos casos donde la función es constante, que son los casos superiores, D1 y D3 están apagados, y D2 y D4 están encendidos por lo que esto representa el tipo de función que se seleccionó en el dip switch. En cualquier otro caso presentado en la figura 3se observa el hecho de que ningún otro caso tiene esta misma configuración de LEDs por lo que entonces la FPGA emulo perfectamente el algoritmo de Deutsch aplicado al problema de Deutsch-Jozsa.

```
NET "A" PULLUP;
NET "A" LOC = F17;
NET "B" PULLUP;
NET "B" LOC = F18;
NET "C" PULLUP;
NET "C" LOC = E16;
NET "D" PULLUP;
NET "D" LOC = D17;
NET "E" PULLUP;
NET "E" LOC = C18;
NET "F" PULLUP;
NET "F" LOC = C17;
NET "G" LOC = P15;
NET "H" LOC = P16;
NET "I" LOC = N15;
NET "J" LOC = N16;
```

Figura 2. Asignación de pines para algoritmo de Deutsch-Jozsa.

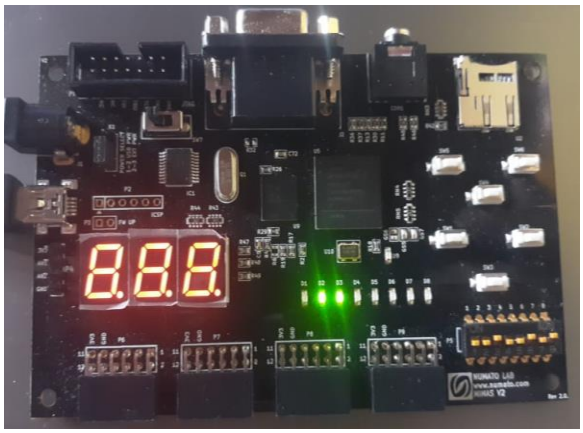


Figura 3. Implementación en FPGA para algoritmo de Deutsch-Jozsa

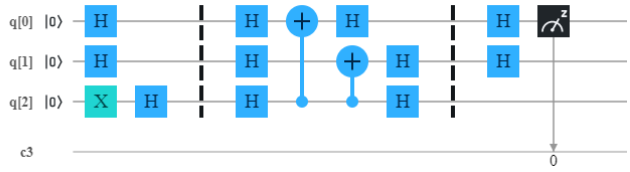
VII. Circuitos cuánticos en IBMQX

Los algoritmos de Deutsch-Josza, Grover y Shor son los algoritmos cuánticos más conocidos. Esto se debe a que estos algoritmos muestran el potencial real que se puede lograr en la computación cuántica. En el presente artículo, estos algoritmos se implementarán en IBM Quantum Experience para estudiar los tiempos de respuesta logrados en un procesador cuántico superconductor real. Estos resultados se compararán con los resultados simulados para analizar el ruido. Los circuitos cuánticos utilizados en los algoritmos de Grover y Shor se basan en implementaciones previamente documentadas [20] [21].

El primer circuito cuántico que se implementará es el algoritmo Deutsch-Jozsa con una función de oráculo equilibrada. El circuito cuántico equivalente para esta implementación en IBM Quantum Experience se encuentra en la figura 4.

El arreglo propuesto debería generar un uno en el Qubit menos significativo basado en los modelos matemáticos para este algoritmo. Las superposiciones iniciales aplicadas a través de las puertas de Hadamard permiten un cierto nivel de paralelismo. Este paralelismo hace que el algoritmo sea capaz de identificar la función oráculo en una sola iteración.

La segunda implementación será el algoritmo propuesto por Lov Grover, para buscar en un registro en una base de datos sin clasificar. El circuito cuántico propuesto implementado en IBMQX se encuentra en la figura 5.



.

Figura *4. Circuito de Deutsch-Jozsa balanceado*

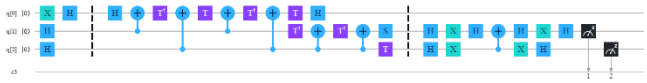


Figura *5. Algoritmo de búsqueda de Grover*

Al igual que en el algoritmo de Deutsch-Jozsa, los Qubits de entrada se inducen en superposición a través de compuertas Hadamard, lo que potencia la velocidad de este algoritmo. La segunda etapa de este algoritmo es la inversión de fase, que invierte el signo del índice correcto, y como último paso, se implementa una inversión sobre la media para resaltar además la respuesta correcta.

El último algoritmo que se implementará es el algoritmo de Shor. Este algoritmo no se puede implementar únicamente en un entorno cuántico, ya que posee una interconexión cuántica y clásica [22]. La etapa cuántica de este algoritmo es responsable de la generación de la muestra, que será útil para encontrar un factor r. El circuito cuántico propuesto para esta implementación es la figura 6.

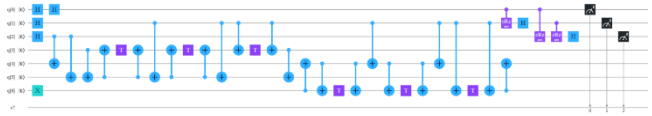


Figura *6. Algoritmo de Shor para factorizar el numero 15*

Este algoritmo utiliza tres Qubits de control y cuatro Qubits de destino. Utiliza la superposición de Qubits de control para realizar las operaciones de álgebra modular. Como etapa final, hay una transformada cuántica de Fourier inversa para generar las muestras obtenidas a través del proceso.

VIII. RESULTADOS

Todos los cálculos realizados en el artículo actual se simularon en IBM Quantum Experience y corresponden a un total de 1024 muestras. Es importante señalar que un mayor número de muestras dará un resultado más preciso.

El primer algoritmo implementado fue el algoritmo Deutsch-Jozsa, ejecutado en el simulador y procesadores cuánticos de IBM. La Tabla 2 muestra los resultados obtenidos para este algoritmo.

Tabla 2. Resultados simulados para el algoritmo de Deutsch-Jozsa

Resultados simulados		
Resultado	Probabilidad	Tiempo de ejecución
0	100%	4ms

Los resultados obtenidos para un entorno simulado fueron correctos las 1024 veces que se ejecutó, y el tiempo de ejecución fue de 4ms. La Tabla 3 muestra los resultados obtenidos para los procesadores cuánticos reales.

Tabla 3. Resultados obtenidos para procesadores cuánticos reales en el algoritmo de Deutsch – Josza

Resultado de procesadores reales				
Resultado	Nombre de procesador			
	Burlington	Melbourne	Ourense	Yorktown
0	40.43%	34.18%	11.33%	3.32%
1	59.57%	65.82%	88.67%	96.68%
Tiempo ejecuc	7.3s	6.4s	6.4s	4.9s

Los resultados obtenidos varían significativamente en comparación con los datos del simulador. Los datos obtenidos tuvieron una precisión media del 77,68% y un tiempo de ejecución de 6,25 s. Las diferencias en los resultados se deben al ruido cuántico y la decoherencia inherente a los procesadores cuánticos, siendo el procesador de Yorktown el más preciso y rápido para esta implementación. En general, los resultados obtenidos fueron correctos la gran mayoría de las veces. Estos resultados confirman que, efectivamente, la implementación se ejecutó con éxito, incluso con presencia de ruido.

El siguiente algoritmo en la cola es el algoritmo de Grover. Este circuito cuántico fue implementado con las mismas condiciones que el de Deutsch-Jozsa. Los datos obtenidos de estas ejecuciones se muestran en la Tabla 4.

Tabla 4. Resultados simulados para algoritmo de Grover

Resultado simulad		
Resultado	Probabilidad	Tiempo de ejecución
110	100%	4ms

Los resultados obtenidos para este cálculo son muy similares a los obtenidos en el algoritmo anterior, precisión perfecta para las 1024 muestras y un tiempo de ejecución dado en ms. Los resultados obtenidos en los procesadores cuánticos reales se muestran en la Tabla 5.

Los datos obtenidos para este cómputo fueron mixtos, con un promedio de 64.63% y un tiempo de ejecución de 6.17s. El ruido obtenido en esta etapa es sustancialmente más alto que los resultados obtenidos en el algoritmo de Deutsch-Jozsa. El ruido excesivo se debe a varios factores como la simplificación del circuito y las tasas de error de las compuertas. Es notable que el procesador más preciso para este algoritmo fue el de Ourense, mientras que el algoritmo Deutsch-Jozsa se ejecutó con mayor precisión en Yorktown, entonces, ¿dónde está la diferencia? Para responder a esta pregunta, debemos remitirnos a los archivos de calibración proporcionados por IBM.

Tabla 5. Resultados obtenidos para procesadores cuánticos reales en el algoritmo de Grover

Resultados de procesadores reales				
Resultado	Nombre de procesador			
	Burlington	Melbourne	Ourense	Yorktown
0	15.82%	8.20%	2.73%	6.06%
10	16.70%	16.80%	7.42%	13.57%
100	17.09%	21.38%	8.98%	6.83%
110	50.39%	53.61%	81%	73.54%
Tiempo ejecución	7.3s	6.4s	6.1	4.9s

Los datos obtenidos para este cómputo fueron mixtos, con un promedio de 64.63% y un tiempo de ejecución de 6.17s. El ruido obtenido en esta etapa es sustancialmente más alto que los resultados obtenidos en el algoritmo de Deutsch-Jozsa. El ruido excesivo se debe a varios factores como la simplificación del circuito y las tasas de error de las compuertas. Es notable que el procesador más preciso para este algoritmo fue el de Ourense, mientras que el algoritmo Deutsch-Jozsa se ejecutó con mayor precisión en Yorktown, entonces, ¿dónde está la diferencia? Para responder a esta pregunta, debemos remitirnos a los archivos de calibración proporcionados por IBM.

Las figuras 7 y 8 contienen el esquema de Qubit y las tasas de error para ambos procesadores. De igual forma, los circuitos cuánticos para Deutsch-Jozsa y el algoritmo de Grover se aprecian en las Ilustraciones. 2 y 3. Es notable que el algoritmo Deutsch-Jozsa posee solo 2 compuertas CNOT, mientras que el algoritmo de Grover posee 7. El procesador cuántico Ourense tiene una tasa de error mínimo para las

compuertas CNOT de 7.073×10^{-3} , mientras que el de Yorktown es 1.423×10^{-3}, por lo tanto, un circuito cuántico compuesto por una mayor cantidad de puertas CNOT se ejecutará con mayor precisión en el procesador de Ourense.

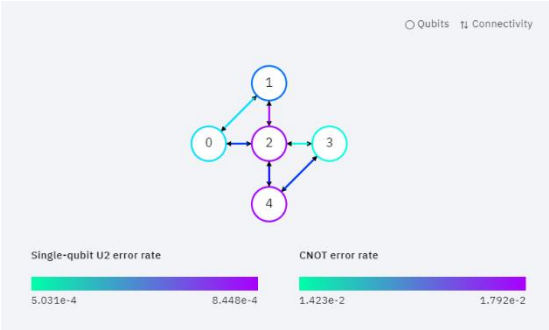


Figura *7. Calibration scheme for IBM Ourense quantum processor.*

Figura *8. Calibration scheme for IBM Yorktown quantum processor*

El último algoritmo a implementar es el algoritmo de Shor para factorizar el número 15. Al igual que los algoritmos anteriores, se ejecutó en un entorno simulado y un procesador cuántico real. Este algoritmo utiliza siete Qubits, por lo que fue ejecutado en el simulador y el procesador cuántico Melbourne de IBM. Los resultados se muestran en la Tabla 6 y la Tabla 7.

Tabla 6. Resultados simulados para el algoritmo de Shor

Resultado simulado		
Resultado	Probabilidad	Tiempo de ejecución
0000000	26.86%	7 ms
0000010	26.07%	
0000100	24.02%	
0000110	23.04%	

Tabla 7. Resultados en el procesador de Melbourne

Resultados de Melbourne		
Resultado	Probabilidad	Tiempo de ejecución
0000000	25.10%	6.7s
0000001	4.10%	
0000010	18.75%	
0000011	4.20%	
0000100	22.66%	
0000101	2.83%	
0000110	18.65%	
0000111	3.71%	

Los resultados para un procesador cuántico real y el resultado simulado fueron muy similares, con una cantidad bastante pequeña de ruido en el procesador. Este resultado es la medida de la transformada cuántica de Fourier inversa, que muestra los períodos obtenidos por la exponenciación del álgebra modular. Estos no son los factores de 15, pero se consideran la parte más difícil de factorizar un número en computadoras clásicas. Para encontrar los factores, los resultados obtenidos por la transformada son introducido en la ecuación $p = \frac{\tilde{x}}{2^L}$ en donde $\tilde{x}$ es el valor obtenido por la transformada, y L es el número total de bits de control del circuito. Aplicado a este caso, el resultado sería: $0, \frac{1}{4}, \frac{2}{4}, \frac{3}{4}$, en donde 4 es el denominador común r más pequeño que cumple la condición $x^r \equiv 1(mod\ n)$. Una vez se ha encontrado el valor de r, se encuentran los factores mediante el máximo común divisor $gcd(a^{\frac{r}{2}} \pm 1, C)$. Para este caso en particular se obtuvo como resultado 3 y 5, que son efectivamente los factores del número 15.

IX. Discusión

Los resultados obtenidos en los algoritmos implementados en la plataforma IBM Quantum Experience coincidieron con los resultados teóricos y la implementación en el FPGA. Existe un antecedente de la ejecución del algoritmo de Deutsch-Jozsa en IBMQX en [23]. El autor no proporciona los tiempos de ejecución para su implementación, sin embargo, los resultados proporcionados confirman la presencia de ruido. Este algoritmo se ejecutó en ibmqx4, que es un procesador cuántico ubicado en Tenerife, pero no disponible a la fecha, por lo que no se logró una comparación directa en precisión.

De igual forma existen registros de la implementación del algoritmo de Grover en un entorno IBMQX en [24]. Este circuito consta de 33 compuertas, misma cantidad que el ejecutado en el presente documento. Los resultados de [24] mostraron una precisión del 59,69% y un tiempo de ejecución de 84,33 segundos para un total de 8192 muestras. Además, [21] implementa el mismo circuito para el algoritmo de

Grover afirmando tener una precisión del 65%. En los últimos años se ha descubierto que el algoritmo de Grover tiene un gran número de aplicaciones debido a las maravillas logradas mediante la inversión de fase y la amplificación. [25] Muestra un algoritmo propuesto basado en el algoritmo de búsqueda de Grover para factorizar primos y tri-primos. Este algoritmo demuestra ser versátil y bastante útil. Sin embargo, es muy susceptible al ruido. [26] contiene un estudio específico sobre el impacto del ruido en el algoritmo de Grover. Afirma que el algoritmo de Grover es totalmente inaccesible para aplicaciones con más de 11 Qubits y bases de datos de 2048 registros sin un método de control de decoherencia.

El algoritmo de Shor se ha implementado en [27] para factorizar 15,21 y 35 utilizando una versión simplificada del algoritmo de Shor. Además, se puede encontrar una explicación paso a paso completa de los métodos de exponenciación modular de Shor en [28]. Este algoritmo se basa en gran medida en su función de oráculo, que debe cambiarse de acuerdo con cada número específico elegido en la etapa de exponenciación modular. [29] Afirma que las demostraciones previas del algoritmo de factorización de Shor requirieron algunas "optimizaciones", haciendo que el compilador "conozca" la respuesta a encontrar. Estas optimizaciones permiten que el algoritmo cuántico se ejecute en menos Qubits y no se considera una forma general de implementación.

X. Conclusiones

Las computadoras cuánticas pueden superar a los sistemas clásicos pues explotan las propiedades de la mecánica cuántica como la superposición, el entrelazamiento y el túnel cuántico, lo que les permite cierto grado de paralelismo. No existen muchas opciones disponibles abiertas al público para ejecutar algoritmos cuánticos. Esto se debe al hecho de que se deben realizar más optimizaciones en la corrección de errores y la decoherencia para lanzar un sistema cuántico completamente comercial. Además, la cantidad y el tipo de puertas utilizadas en un circuito cuántico tuvo un impacto directo en la cantidad de ruido percibido en los resultados obtenidos. Un circuito cuántico que no esté completamente optimizado con la cantidad mínima de puertas cuánticas necesarias podría no funcionar correctamente debido a una cantidad excesiva de ruido. Los diferentes procesadores tienen diferentes resultados en cuanto a precisión, según su topología de Qubit y las tasas de error de las puertas. Esto no permite que un procesador cuántico sea suficientemente general en la ejecución del algoritmo y, además, confirma que debe aplicarse una técnica de corrección de errores adecuada para que sea sostenible. Los resultados de la calibración deben estudiarse para encontrar el mejor procesador adecuado para cada aplicación. Los tiempos de relajación y desfase aún están por ser objeto de estudio para definir su impacto en los tiempos de ejecución. Muchos algoritmos demostraron ser sustancialmente más eficientes que los algoritmos clásicos, pero aún carecen de una forma general de implementación. Las funciones de Oráculo deben modelarse en cada caso específico y, en algunos escenarios, se implementan funciones para una respuesta en específico. Estas implementaciones optimizadas permitieron que los algoritmos se ejecutaran usando menos Qubits, pero no se pueden usar en un segmento más amplio de aplicaciones.

XI. References

[1] G. E. Moore, "Cramming more components onto integrated circuits," Electronics, vol. 38, 1965.

[2] J. Singh and M. Singh, "Evolution in quantum computing," Proc. SMART -2016 IEEE Conf. ID 39669 5th Int. Conf. Syst. Model. Adv. Res. Trends, 2016, doi: 10.1109/SYSMART.2016.7894533.

[3] D. Castelvecchi, "Quantum cloud goes commercial," Nature, 2017.

[4] S. Boixo et al., "Characterizing quantum supremacy in near-term devices," Nat. Phys. 14, 2018, doi: 10.1038/s41567-018-0124-x.

[5] A. G. Vivekanand Mishra, "A review on quantum computing and communication," 2nd Int. Conf. Emerg. Technol. Trends Electron. Commun. Netw., 2014, doi: 10.1109/ET2ECN.2014.7044953.

[6] A. Montanaro, "Quantum algorithms: an overview," Nat. Quantum Inf., 2016, doi: 10.1038/npjqi.2015.23.

[7] "IBM Quantum Computing." https://www.ibm.com/quantum-computing/.

[8] V. Chappidi and S. Ganguly, "Simulation of NMR implementation of Deutsch-Jozsa algorithm," IEEE Int. Adv. Comput. Conf. IACC, 2014, doi: 10.1109/IAdCC.2014.6779431.

[9] M. A. Nielsen and I. L. Chuang, Quantum computation and quantum information. Cambridge University Press, 2010.

[10] L. K. Grover, "A fast quantum mechanical algorithm for database search," Proc. STOC, 1996.

[11] N. S. Yanofsky and M. A. Mannucci, Quantum computing for computer scientists. Cambridge University Press, 2008.

[12] N. Benchasattabuse, P. Chongstitvatana, and C. Aporntewan, "Quantum rough counting and its application to Grover´s search algorithm," 3rd Int. Conf. Comput. Commun. Syst., 2018, doi: 10.1109/CCOMS.2018.8463331.

[13] K. Kang, "Two improvements in Grover's algorithm," 27th Chin. Control Decis. Conf., 2015, doi: 10.1109/CCDC.2015.7162096.

[14] Z. Sakhi, A. Tragha, R. Kabil, and M. Bennai, "Quantum cryptography based on Grover´s algorithm," Second Int. Conf. Innov. Comput. Technol., 2012, doi: 10.1109/INTECH.2012.6457788.

[15] Y. Li and Q. L. Tong Li, "Design and implementation of an improved RSA algorithm," Int. Conf. E-Health Netw. Digit. Ecosyst. Technol., 2010, doi: 10.1109/EDT.2010.5496553.

[16] M. Hayward, "Quantum computing and Shor´s algorithm." 2012.

[17] W. Buchanan and A. Woodward, "Will quantum computers be the end of public key encryption?," J. CYBER Secur. Technol., 2016, doi: 10.1080/23742917.2016.1226650.

[18] S. Hamdi, S. T. Zuhori, F. Mahmud, and B. Pal, "A compare between Shor´s quantum factoring algorithm and general number sieve," Int. Conf. Electr. Eng. Inf. Commun. Technol., 2014, doi: 10.1109/ICEEICT.2014.6919115.

[19] Z. Cao and L. Liu, “On the complexity of Shor´s algorithm for factorization,” Second Int. Symp. Inf. Sci. Eng., 2009, doi: 10.1109/ISISE.2009.86.
[20] L. M. K. Vandersypen, M. Steffen, G. Breyta, C. Yannoni, M. H. Sherwood, and I. L. Chuang, “Experimental realization of Shor´s quantum factoring algorithm using nuclear magnetic resonance,” Nature, vol. 414, 2001, doi: 10.1038/414883a.
[21] P. J. Coles, S. Eidenbenz, S. Pakin, A. Adedoyin, J. Ambrosiano, and P. Anisimov, “Quantum algorithm implementation for beginners,” 2018.
[22] R. J. Lipton and K. W. Regan, Quantum algorithms via linear algebra. Massachusetts Institute of Technology, 2014.
[23] S. Gangopadhyay, Manabputra, B. K. Behera, and P. K. Panigrahi, “Generalization and Partial Demonstration of an Entanglement Based Deutsch-Jozsa Like Algorithm Using a 5-Qubit Quantum Computer,” Quantum Inf. Process., 2018, doi: 10.1007/s11128-018-1932-8.
[24] A. Mandviwalla, K. Shshiro, and B. Ji, “Implementing Grover´s algorithm on the IBM quantum computers,” IEEE Int. Conf. Big Data, 2018, doi: 10.1109/BigData.2018.8622457.
[25] A. Dash, D. Sarmah, B. Behera, and P. Panigrahi, “Exact search algorithm to factorize large biprimes and a triprime on IBM quantum computer,” 2018.
[26] P. J. Salas, “Noise effect on Grover algorithm.” Depto. tecnologías especiales aplicadas a la telecomunicación, U.P.M, Madrid, 2008.
[27] M. Amico, Z. H. Saleem, and M. Kumph, “An Experimental Study of Shor’s Factoring Algorithm on IBM Q.”
[28] D. Candela, “Undergraduate computational physics projects on quantum computing,” Am. Assoc. Phys. Teach., 2015, doi: 10.1119/1.4922296.
[29] J. A. Smolin, G. Smith, and A. Vargo, “Oversimplifying quantum factoring,” Nature, vol. 499, 2013, doi: 10.1038/nature12290.