UNIVERSIDAD TECNOLÓGICA CENTROAMERICANA

FACULTAD DE INGENIERÍA

PROYECTO DE INVESTIGACIÓN

IMPLEMENTACIÓN DE ALGORITMOS CUÁNTICOS CON FPGA

PREVIO A LA OBTENCIÓN DEL TÍTULO:

INGENIERO EN MECATRÓNICA

PRESENTADO POR:

21541272 DARWIN VALENTÍN LUQUE MADRID

ASESOR: JOSÉ LUIS ORDOÑEZ ÁVILA

CAMPUS: SAN PEDRO SULA; ABRIL, 2020

DEDICATORIA

El presente proyecto es dedicado a todas las personas que confiaron en mí, pero principalmente a mi madre y a mi padre quienes me dieron todo lo que estuvo a su alcance para que me pudiera preparar como un profesional, y quienes me han guiado e inspirado a ser la persona que soy.

AGRADECIMIENTO

Sobre todas las cosas le doy gracias a Dios quien me ha guiado e iluminado a lo largo de mi trayectoria como estudiante, pero sobre todo como persona. A mi madre que a pesar de todas las circunstancias ha estado siempre ahí para apoyarme, guiarme y para sacrificarse para verme convertido en la persona que soy. A mi padre que siempre me dio todo lo que estuvo a su alcance para que me pudiera superar todos los días y que ahora desde el cielo sé que me cuida.

A mis tíos, Karla Madrid, Allan Madrid y Marvin Madrid, quienes estuvieron ahí para mí para apoyarme tanto moral como económicamente. A mis abuelos Valentín Madrid e Isabel Bautista quienes a través de su experiencia y cariño me han enseñado a respetar, ser leal, ser responsable y, sobre todo, a ser un buen ser humano.

A mi novia Bessy Alcerro que me ha apoyado en todo momento. Ella me ha apoyado, comprendido y estado conmigo en los momentos de celebración, pero más que todo en los momentos más complicados.

A mis compañeros y amigos que estuvieron conmigo durante esta etapa como estudiantes con quienes estudiamos, aprendimos y nos apoyamos para poder sobre llevar nuestra carrera de la mejor forma posible, Marlon Delcid, Marvin Lorenzana, Lester Torres, Emerson Isaula, Luis Pineda, Cristian Cisneros, Marcelo Rodríguez, Josué Vargas y Nayra Valle.

EPÍGRAFE

"Si no lo puedes explicar de forma simple, no lo entiendes suficientemente bien."

- Albert Einstein

RESUMEN EJECUTIVO

La computación cuántica es uno de los grandes retos de las ciencias computacionales hoy en día, por lo que en la presente tesis se presenta una implementación en hardware del algoritmo cuántico de Deutsch aplicado a distintos problemas utilizando circuitos digitales tradicionales. Los circuitos se desarrollaron utilizando el software Digital, el cual permite generar el circuito a Verilog para luego sintetizarlo en una FPGA. Esta investigación es de enfoque cuantitativo y posee un diseño experimental, para su desarrollo se basó en el modelo Incremental, comenzando desde una compuerta cuántica hasta la realización del algoritmo propuesto aplicado a dos problemas distintos, uno más complejo que el otro. Como principal resultado se pudieron realizar varias compuertas cuánticas para el desarrollo del algoritmo cuántico donde se destaca la compuerta cuántica Haddamard. Como se mencionó el algoritmo se logró aplicar a dos problemas distintos, el problema de Deutsch y el problema de Deutsch-Jozsa. Finalmente se concluye que es factible por un lado implementar un algoritmo cuántico utilizando circuitos digitales tradicionales y obtener un mejor desempeño, en comparación con una simulación en software, al sintetizar dicho circuito en una FPGA. Demostrando esta herramienta ser capaz de emular estos algoritmos cuánticos demostró también que estos aligeran los procesos y permiten una respuesta más rápida para la resolución de los problemas planteados en comparación de su contraparte clásica.

Palabras clave: computación cuántica; algoritmo cuántico; Digital; Verilog; FPGA; modelo Incremental; compuerta cuántica; problema de Deutsch; problema de Deutsch-Jozsa.

ABSTRACT

Quantum computing has been one of the biggest challenges nowadays for computer science, for which then the given thesis we present an implementation in hardware of Duetsch's quantum algorithm applied to different problems using traditional digital circuits. This circuits where develop on a software named Digital, which allows to generate a digital circuit to Verilog for then synthesize it to an FPGA. This investigation has a quantitative approach and poses an experimental design, for which development is based on an incremental model, beginning on a quantum gate all the way to the quantum algorithm itself, this one being applied to two different cases, one being more complex than the other. As a main result various quantum gates were realized for the development in which the quantum gate Haddamard stands out. The algorithm was applied to two different problems, as mentioned, Deutsch's problem and Deutsch-Jozsa's problem. Finally, we concluded that is feasible in a way to Implement de quantum algorithm using classic digital circuits and obtaining a better improve, in comparison with a Simulation in software, to synthesis such circuit on an FPGA. Showing us that this tool it's capable to emulate such quantum algorithms demonstrating that these too speed things up and allow a much quicker answer for the resolution of the described problems in comparison with its classic counterpart.

Keywords: quantum computing; quantum algorithm; Digital; Verilog; FPGA; incremental model; quantum gate; Deutsch's problem; Deutsch-Jozsa's problem.

ÍNDICE DE CONTENIDO

Tabla de Ilustraciones

Tabla de Formulas

ÍNDICE DE TABLAS

CAPÍTULO 1. INTRODUCCIÓN

La computación y la comunicación cuántica son tecnologías revolucionarias prometedoras que explotan los fenómenos cuánticos como superposición y el entrelazamiento para lograr una ventaja exponencial sobre sus contrapartes clásicas, (Bennett & DiVincenzo, 2000; Jozsa & Linden, 2002). Una de las más bizarras y fascinantes predicciones de la teoría de la mecánica cuántica es que la capacidad de procesamiento de información del universo es más extensa de lo que parece. Como supone la teoría, una colección de objetos cuánticos dentro de una caja cerrada va a proceder a hacer, en general, lo que es físicamente capaz de realizar, todo al mismo tiempo. A este sistema cerrado se le describe como función de onda que para más de unas cuantas partículas es una entidad matemática larga describiendo los estados de la materia y estado más allá la experiencia y la intuición, (Ladd et al., 2010).

En las pasadas décadas ha salido a luz una pregunta muy importante, ¿es posible poder guardar y procesar información en un sistema cuántico? Se la busca una respuesta a esta pregunta debido a que el límite de las computadoras clásicas basadas en la Determinist Turing Machine, o maquina determinante de Turing, está cerca. Las computadoras clásicas ya no están siendo suficiente para la necesidad que hay hoy en día en cuanto a los datos de la web y la capacidad de procesamiento de encriptación. Sin embargo, la falta de estos ordenadores dificulta poder generar algoritmos y funciones, debido a sus características cuánticas, entrelazamiento y superposición, hacen que la tarea de emular estos algoritmos sea complicada por la paralelización que estos requieren, ya que estos sistemas clásicos tienen una naturaleza secuencial. Y es por esto que se busca una alternativa viable para poder emular estos algoritmos cuánticos. Para solucionar esta problemática se propone utilizar una FPGA (Field-Programmable Gate Array) que es un dispositivo con varias compuertas lógicas que sus interconexiones pueden ser programadas para poder formar diferentes circuitos. Circuitos que, dependiendo de la programación, pueden trabajar en paralelo, (Barkalov & Titarenko, 2008).

Para el presente proyecto de investigación se propone emular en la FPGA el algoritmo cuántico, con todas las herramientas que este requiere, bajo dos problemas

distintos como ser el problema de David Deutsch (1985) y el problema de Deutsch-Jozsa (1992).

CAPÍTULO 2. PLANTEAMIENTO DEL PROBLEMA

2.1 PRECEDENTES DEL PROBLEMA

Una nueva disciplina, la ciencia de la información cuántica, ha surgido en las últimas dos décadas del siglo veinte debido a la intersección que ha habido entre la física, las matemáticas, y la informática, (Marinescu y Marinescu, 2012). Los sistemas de mecánica cuántica prometen tener una capacidad de procesamiento y almacenamiento mayor a la de un sistema clásico, y, por esto, puede ser implementada para un tipo de computación más poderosa, (Feynman, 1982). La computación cuántica utiliza varios efectos de la mecánica cuántica como el entrelazamiento y la superposición para proporcionar una aceleración masiva del rendimiento en ciertos tipos de problemas de computación como los datos masivos, la factorización y la encriptación, (Khalid, 2005). En cuanto a la encriptación, una computadora con base de entrelazamiento, en teoría, manipula él envió y recibimiento del bit cuántico de manera más segura y eficiente, y también puede ser utilizado para mandar la llave de encriptación a mayores distancias eficientemente, (Kent et al., 2003). En cuanto a la factorización, existe un algoritmo creado por Peter Shor conocido como Shor's algorithm, o algoritmo de Shor. Este algoritmo de Shor proporciona una forma de factorizar grandes números enteros en tiempo polinómico utilizando una computadora cuántica, una tarea para la que no se conoce ningún método clásico eficiente, (Lu et al., 2007). Y en cuanto a los datos masivos, se encontró que la computación cuántica puede ser utilizada a gran escala para analizar el problema del rápido crecimiento de los datos en la web, (Abhishek Pandey, 2015).

La falta de una computadora cuántica ha hecho que la capacidad de poder analizar y crear nuevos algoritmos sea una tarea difícil. Todos los modelos generales de computación existentes son efectivamente clásicos. Es decir, una especificación completa de su estado en cualquier instante equivale a la especificación de un conjunto de números, todos los cuales son en principio cuantificables. Sin embargo, de acuerdo con la teoría de la física cuántica no existe ningún sistema físico con esta propiedad, (Deutsch, 1985). La complejidad se origina en las mismas propiedades que hacen de este sistema mejor que el sistema clásico, las

propiedades cuánticas de un bit cuántico, como ser la superposición, el entrelazamiento y el paralelismo cuántico.

Debido a las nuevas exigencias y el gran poder que ofrecen las computadoras cuánticas es necesario ir revisando si su funcionalidad es tan buena como la teoría lo dice. Por eso se busca poder emular ciertos algoritmos cuánticos y ver su funcionalidad. Para eso se necesitan varios circuitos específicos trabajando en conjunto y de manera paralela.

2.2 Definición del Problema

Como ya se había mencionado, debido a la falta de una computadora cuántica, la simulación de algoritmos cuánticos en una computadora clásica se utiliza ampliamente para verificar las funcionalidades de los algoritmos cuánticos, (Karafylidis, 2005). Sin embargo, la computación clásica no puede hacer uso de toda la capacidad que puede llegar a tener un algoritmo cuántico debido a su naturaleza secuencial, el paralelismo intrínseco que tienen estos algoritmos cuánticos hace que esta naturaleza sea el mayor impedimento para poder simularlos. Debido a esto no se puede hacer un análisis profundo ni explotación de la capacidad que pueden llegar a tener estos algoritmos. Ya que se estancaría en el problema principal que es la lentitud, a comparación a una computadora cuántica, que tiene una computadora clásica. Para poder hacer el análisis de estos algoritmos se necesitará de una alternativa viable que puede solucionar este problema causado por el paralelismo cuántico, superposición y entrelazamiento.

2.3 Justificación

Este tipo de ordenador cuántico es claramente el siguiente paso en la computación. El poder, que la teoría dice que pueden llegar a brindar estas computadoras, es el motivo que hacen poder analizar estos algoritmos a priori a la invención como tal de estos ordenadores algo importante. Siempre estos estudios previos ayudan a poder saber a qué se enfrenta el futuro y ayudan a la preparación para este tema tan complejo.

2.4 Preguntas de Investigación

1. ¿Se puede implementar una FPGA para la emulación del algoritmo cuántico de Deutsch?
2. ¿Hay una relación entre las características cuánticas y los operadores de las compuertas cuánticas?

3. ¿Habrá un circuito lógico digital equivalente para las compuertas cuánticas que se pueda utilizar para la resolución de los problemas propuestos?
4. ¿Se puede generar los circuitos equivalentes de manera de dejar un esquema y función global para el funcionamiento de los algoritmos cuánticos?

2.5 OBJETIVOS

2.5.1 OBJETIVOS GENERALES

1. Implementar una FPGA para emular el algoritmo de Deutsch aplicado al problema de Deutsch y Deutsch-Jozsa.

2.5.2 OBJETIVOS ESPECÍFICOS

1. Explicar la relación que tienen las características cuánticas, superposición y entrelazamiento, con las compuertas cuánticas.
2. Diseñar los circuitos equivalentes de las compuertas cuánticas para la simulación de la resolución de los problemas propuestos.
3. Estandarizar los circuitos a diseñar para cualquier caso posible.

CAPÍTULO 3. MARCO TEÓRICO

Para la comprensión de la emulación de los algoritmos cuánticos en un FPGA se plantearon las siguientes temáticas: la computación cuántica, de manera de comprender desde la base de la computación hasta llegar a los algoritmos cuánticos y los circuitos cuánticos que conllevan a los algoritmos; el FPGA, para poder entender cómo funcionan estos dispositivos, porque estos dispositivos son ventajosos para la investigación y como se programan; y finalmente definir de una manera general y completa la metodología con la que se estará abarcando para la presente tesis, en este caso se escogió la metodología incremental.

Pero primero se debe saber de dónde nace el concepto de computación cuántica. Como ya se había mencionado los dispositivos van cada vez reduciendo más su tamaño al punto que estos dispositivos van a empezar a tener el tamaño de un átomo de hidrogeno. Estos dispositivos están en un tamaño tan pequeño que ya no se rigen por las leyes de la física convencionales, o clásicas, sino por la mecánica cuántica. Pero, ¿qué es mecánica cuántica?

La mecánica cuántica es más adecuada para la descripción de los fenómenos a escalas atómicas y que de alguna manera son más elegantes y satisfactorio que el esquema clásico, (Dirac, 1981). La historia de la mecánica cuántica comienza con la hipótesis cuántica planteada por Max Planck que habla de que cualquier sistema de radiación de energía atómica puede teóricamente ser divido en un numero discreto de elementos de energía.

Planck planteaba que es necesario conocer la energía ocupada por una N cantidad de resonadores, donde N es un número muy grande, ya que esto sería igual a la radiación de energía de un cuerpo oscuro. También logro plantear la igualdad donde dice que esa energía de los resonadores es igual a una constante P, donde este es un entero muy grande, por una cantidad de elementos de energía, ε, tal que la formula se miraría asi:

$$U_N = P\varepsilon$$

A lado del teorema de Kirchoff de la proporcionalidad de emisión y absorción de poder, la llamada ley de desplazamiento, que fue descubierta y nombrada por W. Wien, provee la contribución más valiosa para la fundación del firme establecimiento de la teoría de la radiación de calor, en la forma dada por M. Thiesen:

$$E \cdot dl = q^5 y(lq) \cdot dl$$

Ecuación 1. Ecuación de la radiación de calor en un cuerpo oscuro.

Donde l es la longitud de onda, $E \cdot dl$ representa la densidad de volumen de radiación del "cuerpo oscuro" dentro de la región del espectro l hasta $l + dl$, q representa la temperatura y $y(x)$ representa una cierta función solo para el argumento x. Debido a esto se puede concluir que, es necesario interpretar a U_N no como continuo, una cantidad infinitamente divisible, sino como una cantidad discreta compuesta de un numero integral de finitas iguales partes, (Planck, 1901). La mayor aportación de Planck fue cuando logro plantear la siguiente formula:

$$\varepsilon = hn$$

Ecuación 2. Formula de Planck.

donde, ε es el elemento de energía, n es la frecuencia a la que vibra los resonadores y h es la famosa constante de Planck.

Sin embargo, para que Planck pudiese llegar a esta hipótesis cuántica tuvieron que pasar ciertos experimentos anteriores. Como ser; el descubrimiento de los rayos de cátodos por Michael Faraday (1838); la declaración de problema de la radiación de un cuerpo oscuro por Gustav Kirchoff (1859-1860); la ley de desplazamiento de Wien planteada por Wilhelm Wien, que dicta que la relación entre una constante, llamada constante de Wien, y la temperatura de un cuerpo oscuro es igual a la longitud de onda producida por los picos de emisión del cuerpo oscuro; y el descubrimiento del efecto fotoeléctrico por Heinrich Hertz (1887).

Después, quien continuo el trabajo realizado por Michael Planck fue Albert Einstein. Mayor la densidad de energía y la longitud de onda de la radiación, lo más útil se vuelve para los principios teóricos que se han empleado, y resultaron ser: para longitudes de ondas pequeñas y pequeñas densidades de radiación, como sea, este principio falla completamente. Dependiendo de la suposición a ser contemplada, cuando un rayo de luz se está propagando desde un punto, la energía no es distribuida continuamente sobre incremento espacial, pero consiste de un numero finito de "energy quanta" que está localizado en puntos en el espacio, moverlo sin dividirlo, y puede ser absorbido o generado

solo como un todo, (Einstein, 1905). La oración ya planteada de Einstein fue considerada la oración más revolucionara planteada por un físico en el siglo veinte. Esa "energy quanta" luego fue nombrada fotón, termino presentado por Gilbert N. Lewis (1926).

La frase mecánica cuántica fue acuñada por el grupo físicos Max Born, Werner Heisenberg y Wolfgang Pauli. El termino mecánica cuántica fue presentado por primera vez por Max Born en su investigación "Zur Quantenmechanik." Más adelante físico y filósofo austriaco Erwin Schrödinger expuso sus investigaciones y propuso su fórmula para la mecánica cuántica.

Para entender la fórmula de Schrödinger, se puede imaginar una partícula con una masa m, restringida a moverse por el eje x, y sujeta a una fuerza especifica $F(x,t)$. Una vez sabiendo que se puede calcular la velocidad ($v = dx/dt$), el momento ($p = mv$), la energía cinética ($K = (1/2)mv^2$), o cualquier otra variable dinámica de interés. ¿Cómo se puede determinar $x(t)$? Pues simplemente se aplica la segunda ley de Newton: $F = ma$. Para un sistema conservado, el único que se debería de considerar y, afortunadamente, el único que ocurre a un nivel microscópico, la fuerza que se puede esperar es la derivada de un función de energía potencial ($v \ll c$), $F = \partial V/\partial x$, y la ley de Newton se ve entonces de la siguiente manera: $m\, d^2x/dt^2 = -\partial V/\partial x$. Todo esto junto con las condiciones iniciales apropiadas, típicamente $t = 0$, determina $x(t)$. La mecánica cuántica le apunta al mismo problema, pero desde una perspectiva algo diferente. En este caso se debe buscar una función de onda, $\psi(x,t)$, de la partícula, y se puede conseguir esto a través de las ecuaciones de Schrödinger, (Griffiths, 1995).

$$i\hbar\frac{\partial\psi}{\partial t} = \frac{\hbar^2}{2m}\frac{\partial^2\psi}{\partial x^2} + V\psi$$

Ecuación 3. Formula de Schrödinger.

La mecánica cuántica describe funciones de ondas y estudia las fuerzas de lo microscópico. Lo interesante son sus propiedades. Como ser la capacidad de una partícula de estar en dos estados simultáneamente, rigiéndose por una probabilidad, y la capacidad de una partícula para cambiar el estado de otro. Estas propiedades pueden mejor la idea de computación que se tiene, propiedades que luego se definirá como superposición y entrelazamiento, respectivamente. Además de que permite trabajar con elementos más

pequeños y tener una mayor densidad de ellos. Es así entonces como se introduce este concepto a la que podría ser la nueva generación de computadoras.

3.1 Computación Cuántica

En esta sección se hablará sobre el origen de la disciplina de la computación cuántica, se plantearán los principios básicos de la computación cuántica, la construcción de las compuertas básicas y la comprensión de los algoritmos cuánticos. Una computadora cuántica es una tal que la interferencia de la mecánica cuántica es aprovechada para realizar computación, (Deutsch, 1985). Entrando a la mecánica cuántica, se sabe inmediatamente que se tiene la habilidad, aparentemente, de predecir posibilidades, (Feynman, 1982).

En 1982 Paul Benioff, un físico estadounidense, publica una investigación titulada "Quantum Mechanical Hamiltonian Models of Turing Machine" donde logra crear una modelo Hamiltoniano de mecánica cuántica para una máquina de Turing. Sin embargo, este modelo sigue siendo un modelo clásico debido al hecho de que se puede simular en una máquina de Turing. En el mismo año Richard Feynman propuso hacer uso del concepto de computación cuántica para poder realizar cálculos matemáticos de manera más rápida. Él fue un paso más adelante y creo una verdadera computadora cuántica, que consiste de un sistema de red de giro con interacciones de vecinos más cercanos que son libremente especificados. Pero Feynman no modelo el mecanismo que permite seleccionar arbitrariamente una ley dinámica. No es hasta 1985 que David Deutsch, físico israelí, logra simular con éxito cualquier computador cuántico en su publicación "Quantum theory, the Church-Turing principle and the universal quantum computer," dando así la apertura para la generación de nuevos algoritmos cuánticos.

3.1.1 Computación e Información

Los números "computables" son números que pueden ser descritos como números reales cuyas expresiones decimales pueden ser calculadas por medios finitos, (Turing, 1937). Esto da entender que la computación es un proceso finito en el tiempo y que está fija, que tiene como uno de sus objetivos principales la transferencia de información, concepto que se definirá mejor más adelante. Para entender la computación hay que conocer el concepto básico o la base de esta temática y es la máquina de Turing.

La forma de pensar en la máquina de Turing es como algo compuesto por tres partes – un elemento de control, una mente o centro de escritura y lectura, y una cinta infinita. La cinta está dividida en una secuencia de cuadrados, cada uno cargando un símbolo de un alfabeto finito. La mente lectora va, en un tiempo dado, a escanear un cuadrado de la cinta. Puede leer el símbolo escrito y, bajo las direcciones del elemento de control, puede escribir un nuevo símbolo y también puede mover un cuadrado a la derecha o a la izquierda. El elemento de control es un dispositivo con una finita cantidad de "estados" internos. En un tiempo dado, la siguiente operación de la maquina es determinar, debido al estado actual de elemento de control y el símbolo que está siendo leído por la mente lectora, la posición del nuevo cuadrado. Esta operación va a consistir de tres partes; primero la impresión de un nuevo símbolo en presente cuadro, el cual, por supuesto, puede ser el mismo símbolo que se acaba de leer; el segundo, el paso de un elemento de control a un nuevo estado, que puede ser el mismo estado anterior; y por tercero, mover la mente lectora a un cuadrado de la derecha o la izquierda, (Shannon, 1956).

Ahora como ya se había mencionado, uno de los puntos de la computación es transferir, o procesar, información, pero, ¿qué es información? El término "información" ha sido utilizado en una variedad de formas organizacionales desde genética estructural hasta cultura. Intentos de incluir información en modelos ecológicos que, como sea, principalmente dependen de nociones de retroalimentación débiles o enigmáticas. En un modelo cognoscitivo, información es considerada ser todo dato en bruto disponible para ser procesado, (Casagrande, 1999). También es importante saber cómo medir esta cantidad de información. Si el número de mensajes en el conjunto es finito entonces este número o cualquier función monótona de este número puede ser considerada como la medida de información producida cuando un mensaje es elegido del conjunto, todas las elecciones siendo igualmente probables, (Shannon, 1948).

Shannon en su artículo investigativo "Mathematical Theory of Comunication" estableció un teorema:

$$I = -K \sum_{i=1}^{n} p_i \log_b p_i$$

Ecuación 4. Formula de Shannon.

Donde I es la cantidad de información, K y b son constantes positivas, n es el número de estados y p_i es la posibilidad de que ocurra cada estado. Si toda los p_i son iguales, $p_i = \frac{1}{n}$ entonces H debe tener un incremento monótono, (Shannon, 1948). Dado esto se puede modificar la fórmula de tal manera que queda:

$$I = -K\sum_{i=1}^{n}\frac{1}{n}\log_b\frac{1}{n}$$

Sabiendo que la sumatoria ocurrirá una n cantidad de veces, que $\frac{1}{n} = n^{-1}$, que $\log_a b^c = c \cdot \log_a b$ y asumiendo que $K = 1$, se puede concluir que:

$$I = \log_b n$$

Ecuación 5. Fórmula de Shannon reducida.

3.1.2 Las Características de los Sistemas Computacionales

Las 3 principales características de los sistemas computacionales son la capacidad de información, que ya fue definido en la sección anterior, la velocidad de computación, que se refiere a la rapidez de cambio entre estados, y la universalidad, poder generalizar los sistemas.

Ahora, ya se había mencionado que la transferencia de información es uno de los objetivos de la computación, pero en síntesis se puede definir el objetivo general, o principal, de la computación como encontrar el valor de alguna función. ¿Cómo se realiza esto? Esto lo hace por medio de algoritmos, pero para poder entender el concepto de algoritmo se tiene que entender la muy aclamada teoría de Church-Turing. Toda función computable de numero natural a número natural es recursiva y computable, por principio, por la máquina de Turing, (Church, 1936; Turing, 1937). Esto se refiere con que cualquier procesamiento de información que lleva de un punto A a un punto B tiene que poder ser computarizado por una máquina de Turing. Esta idea de poder transferir o transformar esta información tiene que seguir una secuencia y una serie de instrucciones. Informalmente, un algoritmo es cualquier procedimiento computacional bien definido que toma algún valor, o conjunto de valores, como entrada y produce otro valor, o conjunto de valores, como salida. Un algoritmo es, por lo tanto, una secuencia de pasos computacionales que transforman la

entrada en la salida. También se podría ver a un algoritmo como una herramienta para resolver problemas computacionales bien específicos, (Cormen et al., 2009).

3.1.3 Generaciones de Computadoras

La tecnología, al igual que todo, evoluciona con el tiempo, o más puntual, las computadoras. La primera generación de computadoras fueron las computadoras de tubos de vacío. Una computadora que maneja los circuitos digitales de la computadora por medio de tubos de vacío. Con estos tubos de vacío se podía controlar el flujo de corriente eléctrica en un gran vacío entre los electrodos en el cual un potencial eléctrico se había aplicado. Sin embargo, estas computadoras generaban una gran cantidad de calor y voluminosas, dado esto se ocupaba un cambio y seguir evolucionando en esta tan resiente ciencia. Entonces es aquí donde nace la nueva generación de computadoras, la segunda generación de computadoras, la computadora de transistor.

Esta computadora trabaja a base de transistores que son dispositivos semiconductores de unión bipolar, ya sea NPN o PNP. El transistor se divide en 3 partes – el colector, emisor y base. La base es la parte del transistor que al estar siendo excitada deja que fluya corriente del colector al emisor o, viceversa, del emisor al colector, dependiendo si es una junta NPN o PNP, respectivamente. Sin embargo, más adelante, y siempre pensando en que la computación está evolucionando y ya en este punto se podría llegar a decir que, de una manera casi exponencial, se vendría a crear la que es la tercera generación de computadoras, los circuitos integrados, que también se le puede referir como microchip.

Los circuitos integrados son un conjunto de transistores MOSFET puestos en una capa semiconductora, comúnmente de silicón. Los circuitos integrados van lideraron a tales maravillas como "casas computadoras" – o por lo menos casas conectadas a una computadora central- control automático para los automóviles, y equipos de comunicación portátiles, (Moore, 1965). La predicción y la gran esperanza que Gordon Moore puso en los circuitos integrados no fue para menos, ya en este punto el precio de una computadora era de menos de 10 dólares, hoy en día se pueden conseguir por centavos de dólar, cuando el proyecto de la computadora de tubos de vació fue de miles y miles de dólares. Para 1975 el precio de los circuitos integrados solo iba de caída mientras que para ese año se llegó a tener hasta 5,000 transistores en una sola placa. Pero sus predicciones las vinieron a cumplir

la siguiente generación de computadoras, los microprocesadores. Los microprocesadores son procesadores de computación que integran funciones a su unidad de procesamiento central en una solo placa de circuito integrado, incluso pueden llegar a tener hasta 8 circuitos integrado. Hoy en día el microprocesador comercial con mayor cantidad de transistores tiene 39.54 billones de MOSFETs, (Mujtaba, 2019).

Ahora, la nueva generación de computadoras está cada vez más cerca, esto se refiere a las computadoras cuánticas. Según la tecnología avanza también se reduce el espacio físico que los transistores o dispositivos ocupan. Llegando a un espacio tan reducido que estos ya no se ven afectados por las leyes de la física convencionales o clásica, y entran al mundo la mecánica cuántica. Por ende, cada concepto que se conoce de la computación como el bit y compuertas lógicas, ahora se le conocería como bit cuántico, y las compuertas cuánticas. Estas ya no comportándose ni viéndose como su contraparte clásica. Se estará hablando más a detalle de cada una más adelante. El bit cuántico ahora tiene unas características que lo vuelven más interesante y complejo al mismo tiempo. El entrelazamiento y la superposición características que hacen que el bit cuántico ya no tenga la misma naturaleza secuencial que tiene el bit clásico y ahora tenga un paralelismo y pueda estar en dos estados al mismo tiempo. Permitiéndole poder transferir una mayor cantidad de información que un simple bit clásico, también dificulta la simulación atreves de un sistema clásico debido a la naturaleza secuencial.

3.1.4 Información. Clásica vs. Cuántica

Para entender la diferencia entre a información clásica y la información cuántica hay que comprender los modelos de cada uno de ellos. Se puede empezar por el modelo más sencillo, el modelo clásico. Para ellos se puede utilizar un ejemplo, el motor de Szilard. Este es un sistema que no fue implementado físicamente, fue diseñado como un experimento por un físico húngaro, Leo Szilard en 1929. El sistema funciona de la siguiente manera, se tiene una cámara vacía con una división, una división removible. La cual, claramente, divide el sistema en dos partes en las que los se le llamaran estado 0 y 1. Se puede imaginar una partícula de gas en la cámara en una temperatura T. Por el tamaño de la cámara, hay unos pistones que pueden ser movidos dentro de la cámara, pero no hasta fuera por unos límites que tiene en cada extremo.

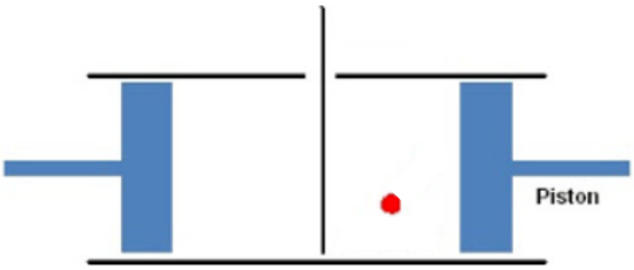


Ilustración 1. Motor de Szilard.

Fuente de la imagen: (McCulloch, 2015).

Como se había mencionado ya, se tienen dos estados debido a las divisiones del sistema, la parte de la izquierda se llamará estado 0 y la parte de la derecha se llamará estado 1. Entonces este sistema puede almacenar un bit de información. Cuando la división es removida el sistema no es un sistema computacional, porque no se puede distinguir los estados.

Ahora se puede imaginar a la partícula de gas en el estado 1. Ahora si se quiere asignar al estado un valor de 0 lo que se tiene que hacer es remover la división y comprimir la cámara de manera que el pistón de la derecha este en medio y luego se pone de vuelta la división en la posición en la que ya estaba. De esta manera se asegura que la partícula se encuentro del lado izquierdo de la división, y, por ende, en estado 0. Sin embargo, esto tomo algo de esfuerzo. Este esfuerzo es igual a una constante, la constante de Boltzamnn, la temperatura del gas y el logaritmo de la razón entre los volúmenes.

$$T = k_B T ln \frac{V_1}{V_2}$$

Ecuación 6. Trabajo realizado al comprimir el motor de Szilard.

Esto es interesante ya que al querer asignar un valor se tiene que disipar algo de energía. Pero más adelante se verá que para cambiar de estado puestos en este punto ya no es necesario disipar energía al menos que se quiera poner el sistema en general como en su estado inicial. En el cual el V_2 sería mayor que el V_1. Se disiparía la misma energía, pero negativamente, lo que dice el signo es si se está expandiendo o comprimiendo el sistema.

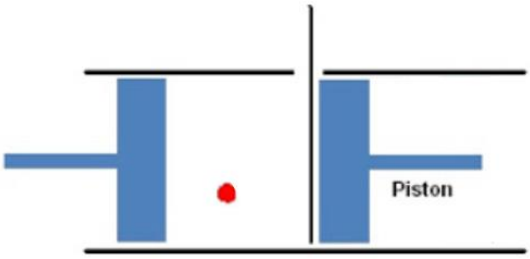


Ilustración 2. Motor de Szilard comprimido hacia la izquierda.

Fuente de la imagen: (McCulloch, 2015).

Aplicando la Formula 3 bajo las condiciones ya planteadas para cambiar de estado el sistema, específicamente de 0 a 1 cuando el sistema empezó en estado 0. Se obtiene que el trabajo realizado, donde k_B y T son constantes dependientes de las condiciones del sistema, fue de:

$$T = k_B T ln2$$

Ahora se tiene este sistema que tiene asignado un estado 0. Se intenta aplicar una compuerta NOT para mapear de estado a 0 ha estado 1. Para esto se va a remover la división, nuevamente, y se va a mover todo el sistema hacia la derecha.

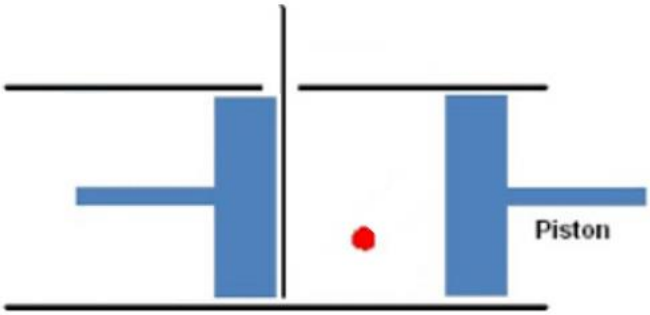


Ilustración 3. Motor de Szilard comprimido hacia la derecha.

Fuente de la imagen: (McCulloch, 2015).

Viendo el sistema de esta manera se puede observar que no hubo un cambio de volumen por ende la razón es igual a 1 y el $ln1 = 0$. Esto da entender que, para cambiar de estado, de 0 a 1, no se disipa energía.

Una vez conocido el modelado del sistema clásico se puede proceder a definir el modelo cuántico. Para el modelo cuántico se va a tomar de referencia una onda

electromagnética, ya que al bit cuántico se le compara con una función de onda senoidal. Como se puede recordar de la clase de física las ondas electromagnéticas, que son ortogonales al eje de propagación.

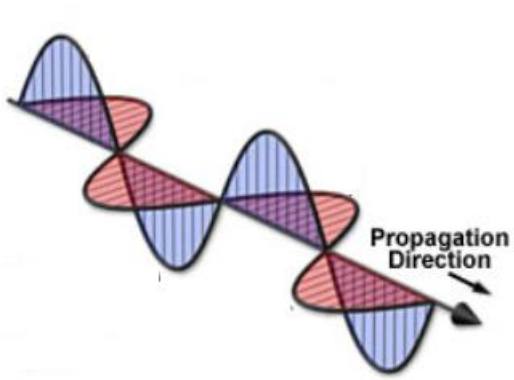


Ilustración 4. Forma de onda del modelado de un bit cuántico.

Fuente de la imagen: (Abramowitz, 2015)

En la ilustración 4 se puede notar que el modelo cuántico tiene una gran similitud con las ondas electromagnéticas, es por esto que se dice que el bit cuántico es una onda de forma; se hace referencia a una superposición y en el caso de más de un bit cuántico se dice que están en entrelazamiento. La información que guarda los bits cuánticos lo codifica por medio de la polarización de un fotón. Matemáticamente el bit cuántico es un vector unitario en un espacio de Hilbert de 2 dimensiones. Las verdaderas partes de las coordenadas en el ejemplo de la Ilustración 4 se puede decodificar en el ángulo de la polarización del fotón mientras las partes imaginarias se pueden observar desde un plano ortogonal al eje de propagación.

$$|\varphi\rangle \in H, \quad \big||\varphi|\big| = 1, \quad \dim H = 2$$

Ecuación 7. Definición de un bit cuántico en un espacio de Hilbert.

Donde φ puede representar un bit cuántico, esta nueva notación se estará explicando más adelante.

3.1.5 Bit Cuántico

La computación cuántica y la información cuántica están construidas en base a un concepto análogo, el bit cuántico. Por propósitos de simulación, el estado del bit cuántico será representado matemáticamente por un vector en un complejo de una dimensión finita,

llamada espacio de Hilbert, (Nielsen y Chuang, 2010). Una de las propiedades más importantes del espacio de Hilbert es que cualquier vector perteneciente al espacio tienen un bien definido producto interno, (Khalid, 2005). Entonces se puede definir para un vector f perteneciente a un espacio de Hilbert la norma:

$$f = \sqrt{\langle f, f \rangle}$$

Ecuación 8. Norma de un vector en un espacio de Hilbert.

Para estar denotando a los bits cuánticos se utilizó la notación de Dirac, la notación bra-ket. La notación ket, que se ve $|x\rangle$, que representa un vector con n cantidad de filas. Donde este vector es parte de un espacio de Hilbert de n dimensión. Y la notación bra, que se ve $\langle x|$, que representa una línea de conjugados con n cantidad de columnas. Y representa un elemento en el espacio dual.

$$|x\rangle = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} \quad \text{y} \quad \langle \text{x}| = (x_1^* \quad x_2^* \quad \dots \quad x_n^*)$$

Ecuación 9. Notación de Dirac.

Donde,

$$|x\rangle \in \text{H} \quad \text{y} \quad \langle \text{x}| \in H^{*1}$$

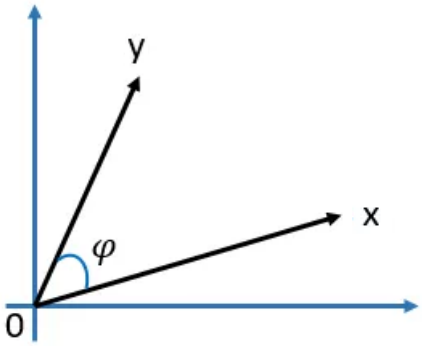


Ilustración 5. Dos vectores en un espacio de Hilbert.

Fuente de la imagen: Elaboración propia.

Esta notación tiene algunas propiedades que van a funcionar más adelante para el control de los bits cuánticos, como ser el producto interno, o también conocido como escalar, entre los vectores está planteado por la siguiente formula:

$$\langle x|y\rangle = \sum_{i=0}^{n} x_i^* y_i$$

Ecuación 10. Producto escalar de dos vectores en un espacio de Hilbert.

La simetría entre los vectores:

$$\langle x|y\rangle = \langle y|x\rangle$$

Ecuación 11. Simetría entre los vectores

La propiedad distributiva entre vectores:

$$\langle x|\alpha y + \beta z\rangle = \alpha\langle x|y\rangle + \beta\langle x|z\rangle$$

Ecuación 12. Propiedad distributiva entre vectores

Los productos internos permiten definir ángulos entre vectores. Los ángulos formados entre los dos vectores se rigen por la siguiente formula:

$$\frac{\langle x|y\rangle}{||x|| \cdot ||y||} = cos\varphi$$

Ecuación 13. Ángulo entre vectores.

En espacios de Hilbert todos los ángulos varían de 0 a $\frac{\pi}{2}$. Pero de debido a que los bits cuánticos son vectores unitarios, la formula ser puede simplificar:

$$\langle x|y\rangle = cos\varphi$$

Ecuación 14. Fórmula de ángulos entre vectores reducida.

Y si el ángulo que se forma entre los dos vectores es de 90°, ósea que los dos vectores son ortogonales, se cumple la siguiente propiedad:

$$\langle x|y\rangle = cos\varphi \rightarrow |x\rangle \perp |y\rangle \rightarrow \langle x|y\rangle = 0$$

Ecuación 15. Propiedad de vectores ortogonales.

Conociendo todas estas propiedades ya se puede plantear lo que es un bit cuántico. Se va a describir a un bit cuántico como un objeto matemático con ciertas propiedades. A pesar de que un bit cuántico es un sistema físico la mayoría de veces se van a estar tratando como un objeto matemático abstracto. La belleza detrás de tratar un bit cuántico como una entidad abstracta es que da la libertad de construir teorías generales de computación

cuántica e información cuántica que no dependerán de un sistema específico para poder ser realizados, (Nielsen y Chuang, 2010).

El bit cuántico puede ser representado como un vector en este espacio de Hilbert como se muestra en la ilustración 6. También se puede observar que se generan dos vectores nuevos que son la base de los cálculos. Que son los vectores $|0\rangle$ y $|1\rangle$, ambos siendo los límites del espacio de Hilbert.

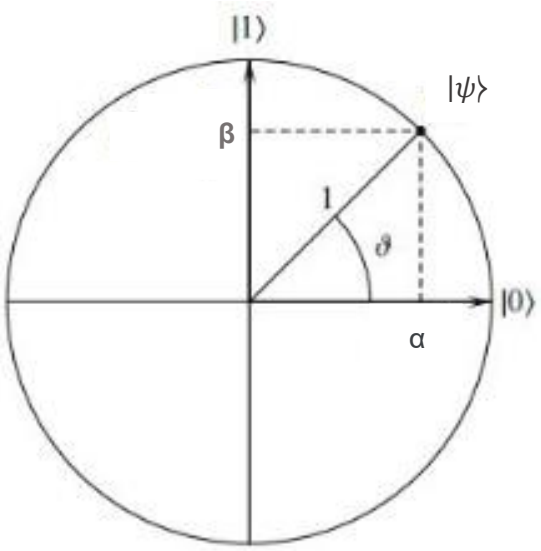


Ilustración 6. Bit cuántico en un espacio de Hilbert.

Fuente de la imagen: (Singhal, 2016)

Viendo la Ilustración 6 se puede, entonces, plantear las bases, los vectores $|0\rangle$ y $|1\rangle$.

$$|0\rangle = \begin{pmatrix}1\\0\end{pmatrix} \qquad y \qquad |1\rangle = \begin{pmatrix}0\\1\end{pmatrix}$$

Ecuación 16. Estados $|0\rangle$ y $|1\rangle$.

Para un bit cuántico el estado es representado por la siguiente ecuación:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$$

Ecuación 17. Ecuación de un bit cuántico

Donde α y β son escalares complejos que están relacionados mediante la siguiente relación:

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

Ecuación 18. Relación entre los escalares α y β

Debido a la Fórmula 17 planteada para el bit cuántico se puede decir que este está en una superposición. La habilidad de un bit cuántico de estar en estado de superposición corre en contra del "sentido común" sobre el mundo físico que rodea todo, (Nielsen y Chuang, 2010). Los estados $|0\rangle$ y $|1\rangle$ de un bit cuántico pueden ser pensados como algo análogo con respecto al bit clásico. Físicamente, los estados $|0\rangle$ y $|1\rangle$ se refieren a una particular polaridad, o giro, de orientación del bit cuántico. Otra forma de ver un bit cuántico es a través de una común pictórica noción conocida como la Bloch Sphere, o la esfera de Bloch.

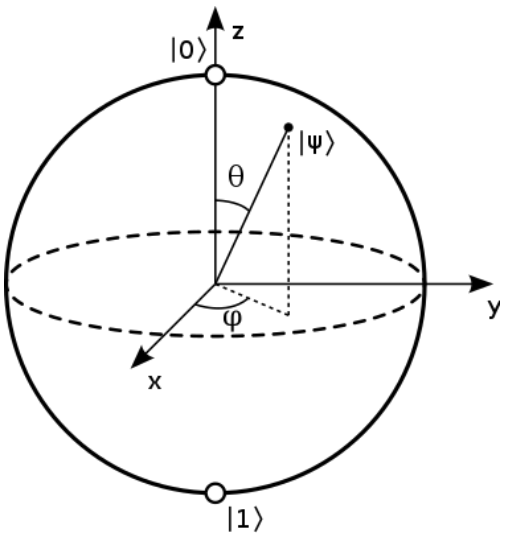


Ilustración 7. Esfera de Bloch.

Fuente de la imagen: (Marinescu y Marinescu, 2012).

Debido a la esfera de Bloch se puede plantear la ecuación de un bit cuántico de la siguiente manera:

$$|\psi\rangle = cos\frac{\theta}{2}|0\rangle + e^{i\varphi}sen\frac{\theta}{2}|1\rangle$$

Ecuación 19. Ecuación de bit cuántico modificada por la esfera de Bloch

Las constantes α y β fueron transformada de coordenadas cartesianas a coordenadas polares sobre un campo de números complejos, y de aquí salen los ángulos θ y φ. Pero desde un punto de vista de simulación, solamente los valores α y β necesitan ser almacenados y manipulados ya que ellos representan la información guardada por los bit cuánticos, (Khalisd, 2005).

Dejando un lado lo extraño, los bits cuánticos son reales, su existencia y comportamiento están validados por experimentos, como el experimento de Stern-Gerlach que provee evidencia de la existencia de los bits cuánticos en la naturaleza, y varios otros sistemas físicos que puede ser utilizado para realizar bits cuánticos. Para tener una idea concreta de cómo se puede realizar un bit cuántico puede ser útil enlistar las formas en las que puede ocurrir. Como las dos formas de polarizar un fotón, como el alineamiento de un giro nuclear un campo magnético uniforme, y como los dos estados de un electrón orbitando un simple átomo como se muestra la Ilustración 6. En el modelo del átomo, el electrón puede existir ya se en el estado de "tierra" o en el estado de "excitado," se va a llamar a cada estado $|0\rangle$ y $|1\rangle$ respectivamente. Al resplandecer algo de luz en el átomo, como la energía apropiada y por la cantidad de tiempo necesaria, es posible mover un electro del estado $|0\rangle$ al estado $|1\rangle$ y viceversa. Pero más interesante, al reducir la cantidad de tiempo del resplandor, el electrón que este inicialmente en estado $|0\rangle$ puede moverse a tan solo la mitad entre $|0\rangle$ y $|1\rangle$, hacia el estado $|+\rangle$, (Nielsen y Chuang, 2010).

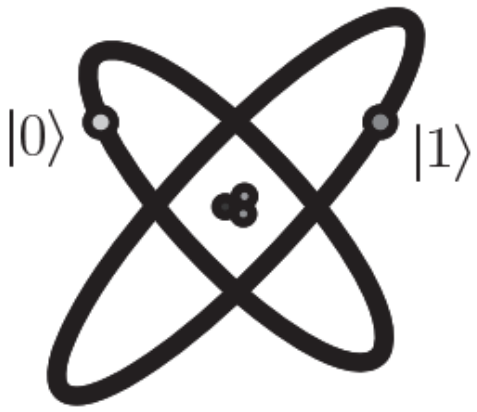


Ilustración 8. Representación de un bit cuántico por dos niveles electrónicos en un átomo.

Fuente de la imagen: (Nielsen y Chuang, 2010)

3.1.6 MEDICIÓN CUÁNTICA

Para empezar, me gustaría abrir la sección con la siguiente frase: la medición altera la función de onda del observador. ¿A qué se refiere esto? Por la Fórmula 14 del bit cuántico se puede deducir que tiene una característica que se llama superposición, más adelante se hablará más en detalle de esta característica, ósea que la computación cuántica es

probabilística. Pero cuando se quiere medir, u observar la función de onda, el sistema siendo medido se destruye y cae en cualquiera de los estados, por eso se dice que medición cuántica es irreversible.

Para entenderlos mejor se puede plantear un caso de un experimento realizado por Thomas Young en 1801, en su publicación "On The Theory of Light and Colours." El experimento de Young, llamado experimento de la doble abertura, es un sistema en el que se plantea un emisor de fotones de un lado, una pared con dos aberturas enfrente del emisor y atrás una gran pared en el que se va a analizar la función de onda dependiendo de los casos que se esté planteando. Para el primer caso se va a plantear el experimento de Young, pero con un sensor de luz en cada abertura.

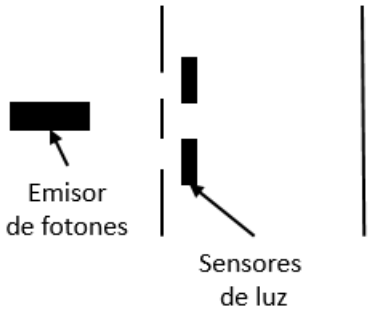


Ilustración 9. Experimento de Young con dos sensores de luz.

Fuente de la imagen: Elaboración propia.

Claro que este experimento se podría ver como un sistema clásico de computación donde el sensor de arriba puede ser el estado 1 y el sensor de abajo el estado 0. Ahora que pasa si se remueve uno de los dos sensores. Lo que se podría observar en la pared trasera es una función de onda que se puede analizar.

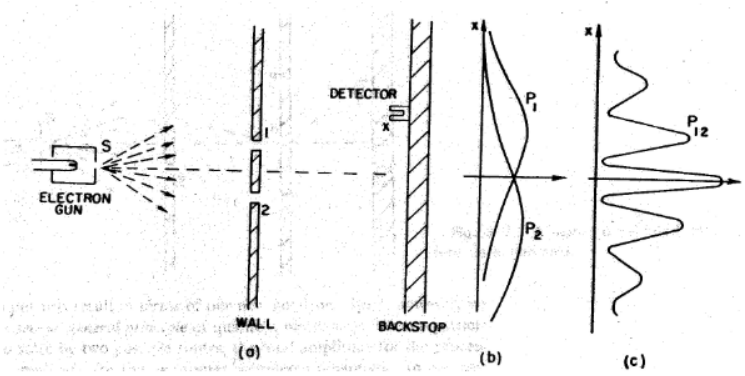


Ilustración 10. Experimento de Young.

Fuente de la imagen: (Young, 1802)

Viendo la Ilustración 10 se puede ver, según las funciones de onda en los tres casos, solo el sensor de arriba, solo el sensor de abajo y sin ninguno de los sensores, se puede concluir que estas ondas y sus desfases se rigen por la posibilidad que tiene cada fotón de elegir una abertura. Y lo más interesante es como la luz al tener dos aberturas modifica la función de onda a una tendencia senoidal amortiguada. En caso que la posibilidad de un estado fuera mayor que la otra la onda se vería más desfasada hacia ese estado.

En aquel momento que Young planteo este ejercicio no se imaginó que esto iba a ser un gran apoyo a la mecánica cuántica y al modelado de una computadora cuántica. El explico este experimento según las leyes de la física que él conocía. Más adelante el experimento realizado por Clinton Davisson y Lester Germer en 1827, denotado como experimento de Davisson-Germer, comprobaron que estas funciones de ondas en realidad se rigen por la mecánica cuántica y se pueden explicar por medio de la Formula 3.

Uno puede estructurar un caso algo ridículo. Un gato es encerrado en una cámara de metal, con un dispositivo diabólico, una sustancia un poco radioactiva que está en un contador de Geiger. La radioactividad es tan pequeña que, probablemente, después de una hora uno de los átomos decae, con la misma posibilidad puede no decaer el átomo. Si ocurre, el tubo del contador se descarga a través de un relé dejando caer un martillo que quiebra un pequeño frasco de ácido cianhídrico. Si uno dejase este sistema por una hora uno puede decir que el gato sigue vivo si por mientras el átomo no se ha decaído. La primera decaída atómica lo hubiera envenenado. La función ψ de todo el sistema expresaría esto teniendo a todo los gatos vivos y muertos mezclados o esparcido en partes iguales, (Schrödinger, 1935). Pero si se abre la caja se va a destruir el sistema ya que la función pasaría de su estado de superposición a encontrar ya sea el gato muerto o vivo, ósea en estado $|0\rangle$ o $|1\rangle$.

Sabiendo la Fórmula 14 del bit cuántico se puede deducir que tiene una característica que se llama superposición, más adelante se hablará más en detalle de esta característica. Debido que el bit cuántico está en superposición, se rige por posibilidades. Y aquí es donde entra uno de los más grandes problemas al momento de modelar y querer hacer uso del bit cuántico y es que al querer medir un bit cuántico este al ser observado se destruye, o cae a

cualquiera de sus dos estados. Sin embargo, estas posibilidades las se puede medir con respecto a los escalares de cada base, α y β.

$$P(|0\rangle) = |\alpha|^2$$

Ecuación 20. Medición de las probabilidades de estado |0⟩.

$$P(|1\rangle) = |\beta|^2$$

Ecuación 21. Medición de las probabilidades de estado |1⟩.

También se puede encontrar una expresión para el ángulo ϑ que se forma entre el bit cuántico $|\psi\rangle$ y la base $|0\rangle$, que se vería de la siguiente manera:

$$\langle 0|\psi\rangle = \cos\vartheta$$

Ecuación 22. Fórmula para encontrar el ángulo entre el bit cuántico y el eje horizontal utilizando el producto interno entre el bit cuántico |ψ⟩ y el estado |0⟩.

De igual forma, se puede calcular la relación que tendría el bit cuántico $|\psi\rangle$ y la base $|1\rangle$, de tal manera que:

$$\langle 1|\psi\rangle = \operatorname{sen}\vartheta$$

Ecuación 23. Fórmula para encontrar el ángulo entre el bit cuántico y el eje horizontal utilizando el producto interno entre el bit cuántico |ψ⟩ y el estado 94

De igual manera se puede resolver el mismo producto escalar reemplazando $|\psi\rangle$ debido a la relación presentada en la Formula 14, de manera que:

$$\langle 0|\psi\rangle = \langle 0\big|\alpha|0\rangle + \beta|1\rangle\rangle$$

Utilizando la propiedad vista en la Formula 9 se puede resolver el anterior sistema de la siguiente forma.

$$\langle 0|\psi\rangle = \alpha\langle 0|0\rangle + \beta\langle 0|1\rangle$$

El producto interno entre el mismo estado $|0\rangle$ es igual a 1 y el producto interno entre el estado $|0\rangle$ y $|1\rangle$ es igual 0, se puede deducir lo siguiente.

$$|\langle 0|\psi\rangle| = \alpha$$

Ecuación 24. Producto escalar entre el bit cuántico |ψ⟩ y el estado |0⟩.

De la misma manera se puede resolver el producto escalar entre el bit cuántico |ψ⟩ y el estado |1⟩:

$$|\langle 1|\psi\rangle| = \beta$$

Ecuación 25. Producto escalar entre el bit cuántico |ψ⟩ y el estado |1⟩.

Pudiendo entonces relacionar la Fórmula 16, Fórmula 18 y la Fórmula 20, se puede concluir lo siguiente.

$$P(|0\rangle) = |\alpha|^2 = |\langle 0|\psi\rangle|^2 = cos^2\vartheta$$

Ecuación 26. Relación entre probabilidad del estado |0⟩ y el ángulo ϑ.

Sabiendo entonces que también se puede relacionar la Fórmula 17, la Fórmula 19 y la Fórmula 21, se puede concluir lo siguiente.

$$P(|1\rangle) = |\beta|^2 = |\langle 1|\psi\rangle|^2 = sen^2\vartheta$$

Ecuación 27. Relación entre la probabilidad del estado |1⟩ y el ángulo ϑ.

En algún sentido la medición es similar a la digitación, porque a pesar de los infinitos posibles estados debido a la superposición del bit cuántico, se puede distinguir dos posibles estados. La elección de la base, ósea el estado, que se está haciendo es importante porque estas elecciones van a definir los resultados de los algoritmos que más adelante se van a estar analizando. Ahora, se consideran los siguientes casos para realizar la medición de los bits cuánticos, donde $|\psi\rangle$ y $|\varphi\rangle$ son los estados a considerar.

$$|\psi\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$$

$$|\varphi\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$$

Viendo las bases $|0\rangle$ y $|1\rangle$ no se obtiene ninguna información sobre estos estados. Porque si se analiza la posibilidad de que cada base se cumpla por medio de la Formulas 16 y 17.

$$P(|0\rangle|\psi) = P(|1\rangle|\psi) = \frac{1}{2}$$

$$P(|0\rangle|\varphi) = P(|1\rangle|\varphi) = \frac{1}{2}$$

En este caso las bases de Haddamard, $|+\rangle$ y $|-\rangle$, estas van ayudar para poder sacar conclusiones de estos dos estados.

$$|+\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$$

Ecuación 28. Ecuación de Haddamard de la base $|+\rangle$.

$$|-\rangle = \frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle$$

Ecuación 29. Ecuación de Haddamard de la base $|-\rangle$.

Aplicando la fórmula 16 se puede sacar las posibilidades de que los casos ya planteados de bits cuánticos $|\psi\rangle$ y $|\varphi\rangle$ estén en estas nuevas bases planteadas.

$$P(|+\rangle|\,\psi) = |\,\langle\psi|+\rangle|^2 = 1$$

$$P(|+\rangle|\,\varphi) = |\,\langle\varphi|+\rangle|^2 = 0$$

3.1.7 Dos o más Bits Cuánticos

En esta sección, se va describir la representación matemática de un sistema con múltiples bits cuánticos. Se va a representar entonces el estado compuesto por varios sistemas cuánticos como un vector unitario en algún espacio de Hilbert. Para eso se va a introducir la base de este nuevo sistema.

qubit I	qubit II	vector
$\|0\rangle$	$\|0\rangle$	$\|00\rangle$
$\|0\rangle$	$\|1\rangle$	$\|01\rangle$
$\|1\rangle$	$\|0\rangle$	$\|10\rangle$
$\|1\rangle$	$\|1\rangle$	$\|11\rangle$

Tabla 1. Bases para sistemas con dos bits cuánticos.

Fuente de la imagen: (Sysoev, 2019).

En la tabla 1 se presentan los estados de respuesta tras la unión de dos bits cuánticos, se presenta también este sistema como vectores. Vectores que se van a plantear como un producto tensor de los vectores que los conforman.

$$|00\rangle = |0\rangle \otimes |0\rangle = \begin{pmatrix}1\\0\end{pmatrix} \otimes \begin{pmatrix}1\\0\end{pmatrix} = \begin{pmatrix}1\\0\\0\\0\end{pmatrix}$$

Ecuación 30. Resolución del producto de tensor entre dos vectores en un espacio de Hilbert, utilizando el vector |00⟩.

Ahora se van a plantear el resto de los vectores en la tabla de la Ilustración 11 para poder observar los vectores resultantes.

$$|00\rangle = \begin{pmatrix}1\\0\\0\\0\end{pmatrix}, \quad |01\rangle = \begin{pmatrix}0\\1\\0\\0\end{pmatrix}, \quad |10\rangle = \begin{pmatrix}0\\0\\1\\0\end{pmatrix}, \quad |11\rangle = \begin{pmatrix}0\\0\\0\\1\end{pmatrix}$$

Ecuación 31. Vectores de los estados con dos bits cuánticos.

Otra observación importante que se puede hacer con estos vectores es que la dimensión aumento por el doble. Esto ayuda a poder definir la dimensión de los espacios de Hilbert en los que están estos estados.

$$dimH = 2^n$$

Ecuación 32. Fórmula para la dimensión del espacio de Hilbert.

Ahora se va a plantear la siguiente base donde se va a plantear un bit cuántico en estado |0⟩ y otro bit cuántico en superposición.

qubit I	qubit II	vector
$\|0\rangle$	$\alpha\|0\rangle + \beta\|1\rangle$	$\alpha\|00\rangle + \beta\|01\rangle$

Tabla 2. Base para dos bits cuánticos con un bit cuántico en superposición.

Fuente de la imagen: (Sysoev, 2019).

Ahora basados en que este sistema está en un estado de superposición también se puede medir la posibilidad de obtener cada estado planteado, utilizando de base la ecuación 20 y ecuación 21, se puede concluir las siguientes posibilidades.

$$P(|00\rangle) = |\alpha|^2$$

$$P(|01\rangle) = |\beta|^2$$

$$P(|10\rangle) = P(|11\rangle) = 0$$

Ahora se va a plantear los dos bits cuánticos en estado de superposición de manera de poder ver como en realidad van a estar los bits cuánticos en todo momento.

qubit I	qubit II	vector
$\alpha\|0\rangle + \beta\|1\rangle$	$\gamma\|0\rangle + \delta\|1\rangle$	$\alpha\gamma\|00\rangle + \alpha\delta\|01\rangle + \beta\gamma\|10\rangle + \beta\delta\|11\rangle$

Tabla 3. Base para dos bits cuánticos en estado de superposición

Fuente de la imagen: (Sysoev, 2019).

Para estos se puede plantear el estado de la siguiente manera:

$$\alpha\gamma|00\rangle + \alpha\delta|01\rangle + \beta\gamma|10\rangle + \beta\delta|11\rangle \rightarrow \alpha|00\rangle + \beta|01\rangle + \gamma|10\rangle + \delta|11\rangle$$

Ecuación 33. Estado de dos bits cuánticos en superposición.

Debido a la superposición se puede medir la posibilidad de cada base utilizando la ecuación 20, esto hace que la suma de posibilidades debe ser igual a un 100% o 1. Se puede relacionar los cuatro escalares de cada base de manera que cumplan la siguiente relación:

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

Ecuación 34. Relación de escalares de estado de dos bits en superposición.

Lo más asombroso de la forma de describir sistemas de múltiples bits cuánticos es que cualquier vector unitario en espacio construido describe algún sistema cuántico real que puede ser implementado en partículas reales.

Pero no todos los vectores pueden ser construidos como productos de tensores de sistemas más pequeños. Considerando el siguiente ejemplo del siguiente conjunto de estados:

$$\alpha|00\rangle + \beta|11\rangle, \quad \alpha|00\rangle + \beta|01\rangle, \quad |\alpha|^2 + |\beta|^2 = 1$$

Estos son vectores unitarios en un espacio de Hilbert de cuatro dimensiones entonces representan un sistema de 2 partículas. Pero estos vectores no son productos tensores de un vector de dos dimensiones, lo que significa que las dos partículas no tienen sus estados separados. Estos tipos de estados son llamados estados de entrelazamiento, y partículas implementando este estado son partículas entrelazadas, se mencionará más detalle de esta característica más adelante. El sistema compuesto puede ser medido como un simple sistema de un bit cuántico.

Para medir un sistema de dos bits cuánticos se puede medir cada bit cuántico de manera individual. Después de la medición de un bit cuántico, por ejemplo, se puede dejar que la medición sea $|0\rangle$, se obtiene la siguiente descripción del sistema:

$$|0\rangle\left(\frac{\alpha|0\rangle}{\sqrt{|\alpha|^2+|\beta|^2}}+\frac{\beta|1\rangle}{\sqrt{|\alpha|^2+|\beta|^2}}\right)$$

La medición del primer bit cuántico no altera el estado del segundo bit cuántico, lo que es de esperarse, ya que los bits cuánticos se encuentran en partículas distintas, y cuando se mide la primera partícula la segunda permanece sin ser tocada. Ahora si se trata de hacer lo mismo en un sistema entrelazado, se puede observar que la medición de la primera partícula le asigna un estado a la segunda partícula.

La interpretación de muchos mundos de la mecánica cuántica puede describir los sitios sin ninguna noción de acción. Para un estado de no entrelazamiento las dos partículas se tiene 4 medidas distintas, cada una existiendo en un multiverso.

$\|00\rangle$	$\|01\rangle$
$\|10\rangle$	$\|11\rangle$

Tabla 4. Estados de dos bits cuánticos no entrelazados en el multiverso.

Fuente de la imagen: (Sysoev, 2019).

Ahora el multiverso para un estado de bits cuánticos entrelazados se mira de la siguiente manera:

$\|00\rangle$	$\|11\rangle$

Tabla 5. Estado de dos bits cuánticos entrelazados en el multiverso.

Fuente de la imagen: (Sysoev, 2019).

3.1.8 SUPERPOSICIÓN

El principio de superposición es una de las características que hace que el sistema cuántico sea superior, exponencialmente, que le sistema clásico. La Fórmula 17, como ya se había mencionado, esto da a entender que un bit cuántico está en estado de superposición.

El principio de superposición dicta que para cualquier vector $|\psi\rangle$ en un espacio de Hilbert corresponde a una posibilidad de estados; dado dos estados, $|\varphi\rangle$ y $|\xi\rangle$ se puede formar otro estado como una superposición $\alpha|\varphi\rangle + \beta|\xi\rangle$. Esta característica de un espacio de Hilbert, la cual contiene todas las posibles superposiciones de los vectores, es apropiada para describirse como un efecto de interferencia, (Marinescu y Marinescu, 2012). De manera que aplicando esto descrito se puede modificar la Formula 17 de manera que:

$$|\psi\rangle = \alpha|\varphi\rangle + \beta|\xi\rangle$$

Ecuación 35. Formula de un bit cuántico modificada a dos estados cualesquiera.

Sin embargo, para poder llamar un estado un estado de superposición tiene que cumplir ciertas propiedades de transformación lineal:

1. El orden de los estados de los vectores no importa para una superposición, ósea que estos están en simetría, cumpliéndose la siguiente afirmación:

$$|\psi\rangle = \alpha|\varphi\rangle + \beta|\xi\rangle = \beta|\xi\rangle + \alpha|\varphi\rangle$$

Ecuación 36. Simetría entre estados de una superposición

2. Todos los estados en superposición pueden ser representados como superposición de los otros estados, ósea que se pueden despejar como variables:

$$|\varphi\rangle = \frac{1}{\alpha}(|\psi\rangle - \beta|\xi\rangle)$$

Ecuación 37. Representación de una superposición como otra.

3. Una superposición de un estado consigo mismo resulta en el estado original, de manera que solo existe una sola posibilidad:

$$\alpha|\varphi\rangle + \beta|\varphi\rangle = (\alpha + \beta)|\varphi\rangle$$

Ecuación 38. Dos estados iguales en una superposición.

Tal computación, que es la computación cuántica, tampoco tiene porque evaluar funciones mientras está resolviendo problemas, porque el estado de su salida puede ser coherente con la superposición de estados correspondiente a la diferente respuesta, lo cual resuelve el problema, (Deutsch y Josza, 1992). Esto es fascinante, ya que con una computadora cuántica no se tiene que saber que está computando para saber la respuesta

gracias a la superposición, luego se hace referencia de esto como un oráculo, pero se explicará esto más adelante.

3.1.9 ENTRELAZAMIENTO

Se dice que el estado puro de dos bits cuánticos está en entrelazamiento si no pueden ser escritos como un producto de tensor de los estados individuales de los dos bits cuánticos, tal como $|\psi\rangle\otimes|\varphi\rangle$, (Vedral y Plenio, 1998) El entrelazamiento cuántico representa una situación en la que dos o más distintas partículas cuánticas se comportan de tal manera que cuando una de las partículas es sujeta a un cambio, un cambio similar le ocurre a cualquier otra partícula entrelazada, (Khalid, 2005). Gracias a esto se cree que el entrelazamiento es la propiedad más importante e interesante de la mecánica cuántica. Esta propiedad fue vista por primera vez por Albert Einstein, Boris Podolsky y Nathan Rosen cuando, en 1935 hicieron un experimento mental, llamado paradoja de EPR, por las iniciales de los apellidos de cada científico, en el que se dieron cuenta que, en un sistema conformado por dos fotones, los cuales ellos llamaron Alice y Bob, el cambio de giro de un fotón afectaba el giro del otro simultáneamente.

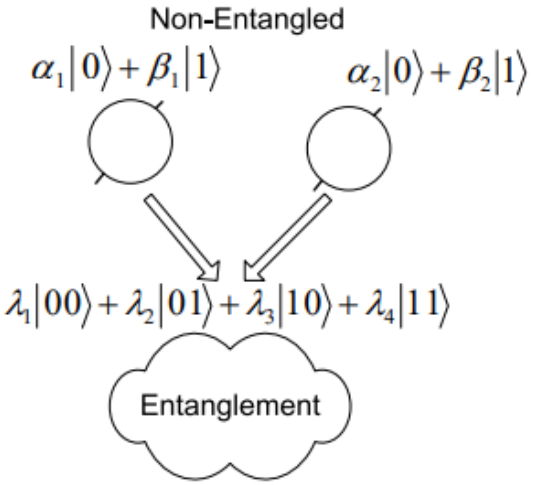


Ilustración 11. Entrelazamiento de dos partículas cuánticas.

Fuente de la imagen: (Khalid, 2005).

Según Albert Einstein (1935), la idea de entrelazamiento cuántico le parecía inusual, ya que no cumplía el principio de localidad y había un gran problema a la hora de querer hacer una medición. Einstein planteo dos preguntas: (1) "¿es la teoría correcta?" y (2) "¿es la descripción da por la teoría completa?" Cualquier elemento de la realidad física debe tener

una contraparte en la física teórica. Empezando por la suposición de que una función de onda, bit cuántico, si da un completa descripción de la realidad física, se llega a la conclusión que dos cantidades físicas, con ningún operador conmutado, puede tener realidades simultaneas. Por lo tanto, la negación a (1) lidera a la negación de su otra única alternativa (2). Lo que lleva a concluir que la descripción que brinda la mecánica cuántica sobre la realidad física dada por las funciones de ondas no es completa, (Einstein, Podolsky, y Rosen, 1935).

En base a esta investigación realizada por Einstein, Podolsky y Rosen, se crea un canal llamado EPR, que, por ejemplo, cuando ciertos tipos de átomos y moléculas decaen con la emisión de dos fotones, y que consisten del hecho de que los dos fotones están siempre encontrados con polarizaciones distintas, independientemente de la base, o estado, utilizado para medirlos, una vez proveídos ambos fotones son medidos con respecto a la misma base, (Gruska, 2000). Debido a este canal se puede llamar a EPR como un estado de un multiverso de bits cuánticos.

$$|\psi_{EPR}\rangle = \frac{(|01\rangle + |10\rangle)}{\sqrt{2}}$$

Ecuación 39. Ecuación del estado EPR.

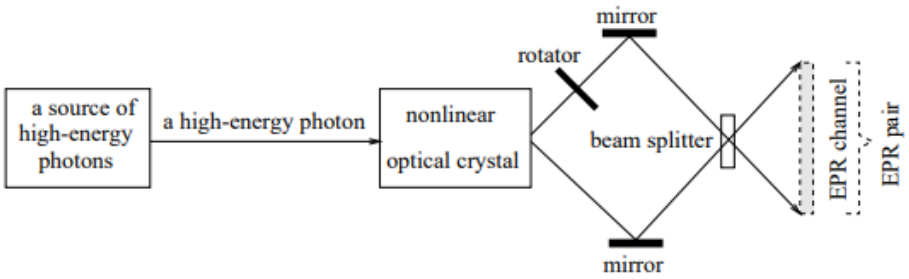


Ilustración 12. Generación de un par de fotones EPR polarizados y un canal de EPR.

Fuente de la imagen: (Gruska, 2000)

El estado EPR no puede ser descompuesto a un producto de tensor de ninguna forma, y por eso se dice que está en entrelazamiento. Los estados $|01\rangle$ y $|10\rangle$ son un producto de tensor de los estados $|0\rangle$ y $|1\rangle$, que se puede observar en la Formula 30, pero su superposición no lo es. Pero la pregunta importante es, ¿cómo se puede crear un sistema de entrelazamiento a partir de un sistema no entrelazado? Esta pregunta se podrá responder más adelante cuando se haga mención sobre la compuerta de Haddamard y la compuerta

cuántica CNOT. Para un adelanto de la respuesta esta será la forma en la que se miraría la resolución:

$$|0\rangle \otimes |1\rangle \rightarrow |01\rangle \rightarrow H \rightarrow (|0\rangle + |1\rangle)|1\rangle \rightarrow CNOT \rightarrow |01\rangle + |10\rangle$$

3.1.10 Compuertas Cuánticos

Primero, para entender las compuertas cuánticas, se tiene que plantear lo que son los operadores unitarios, ya que cualquier compuerta o transformada tiene que cumplir las propiedades de estos operadores. Se va a definir un operador unitario sabiendo las siguientes propiedades.

$$U: H \rightarrow H, \qquad U^*U = UU^* = I$$

Ecuación 40. Propiedad 1 de un operador unitario.

Donde H es un espacio de Hilbert, U es el operador unitario, U^*es el adjunto de U e I es un vector identidad.

$$U: H \rightarrow H, \qquad \big||U|x\rangle\big| = ||x||$$

Ecuación 41. Propiedad 2 de un operador unitario.

Donde $|x\rangle$ es un vector en un espacio de Hilbert. Esta segunda propiedad muestra como estos operadores unitarios conservan módulos.

$$U: H \rightarrow H, |\langle U|x\rangle|U|y\rangle\rangle| = |\langle x|y\rangle|$$

Ecuación 42. Propiedad 3 de un operador unitario.

Donde $|x\rangle$ y $|y\rangle$ son vectores en un espacio de Hilbert. Esta tercera propiedad muestra como estos operadores unitarios conservan los ángulos entre vectores.

El primero operador del que se va a estar hablando es del operador de Haddamard, o la compuerta de Haddamard, cuya denotación es H. Es una forma de transformada de Fourier que se utiliza para hacer rotar un bit cuántico, y mapear los estados $|0\rangle$ y $|1\rangle$ ante una superposición con igual cantidad de peso para ambos estados. Como ya se había planteado las computadoras cuánticas logran su incremento de rapidez a través del hecho que pueden operar múltiples estados al mismo tiempo, que es la superposición, ósea trabajando paralelamente. Esta compuerta es la más utilizada para la transformación

ortogonal de un vector de un espacio de Hilbert. La compuerta cuántica tiene la siguiente estructura:

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

Ecuación 43. La compuerta de Haddamard.

Por lo que la compuerta de Haddamard se comporta con los estados $|0\rangle$ y $|1\rangle$ de la siguiente manera.

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

$$H|1\rangle = \frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix}\begin{pmatrix}1\\ 0\end{pmatrix} = \frac{1}{\sqrt{2}}\begin{pmatrix}1\\ 1\end{pmatrix} = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) = |-\rangle$$

Es interesante ver como esta compuerta vuelve a un vector en estado $|0\rangle$ o $|1\rangle$ en estado de superposición, haciéndolo rotar.

El siguiente operador que se planteará es la compuerta cuántica NOT que se rige sobre la misma lógica que la compuerta NOT convencional que es cambiar de estado un bit. A este operador se denota como X, y se ve de la siguiente manera:

$$X = \begin{pmatrix}0 & 1\\ 1 & 0\end{pmatrix}$$

Ecuación 44. Compuerta cuántica NOT.

Se prosigue a evaluar cómo se comporta la compuerta NOT con los estados $|0\rangle$ y $|1\rangle$.

$$X|0\rangle = \begin{pmatrix}0 & 1\\ 1 & 0\end{pmatrix}\begin{pmatrix}1\\ 0\end{pmatrix} = \begin{pmatrix}0\\ 1\end{pmatrix} = |1\rangle$$

$$X|1\rangle = \begin{pmatrix}0 & 1\\ 1 & 0\end{pmatrix}\begin{pmatrix}0\\ 1\end{pmatrix} = \begin{pmatrix}1\\ 0\end{pmatrix} = |0\rangle$$

La siguiente compuerta que se estará mencionando es la compuerta CNOT, por las siglas en inglés Conditional NOT o NOT Condicional, y se denota $CNOT$. Esta compuerta es un poco más compleja que las otras 2 ya planteadas. Ya que esta compuerta solo se puede plantear para dos bits cuánticos o más. La compuerta funciona de manera de que se tiene un bit de control y un bit controlado, donde el bit controlado cambia de estado si y solo si el bit de control es 1. El operador tiene la siguiente estructura:

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

Ecuación 45. Compuerta cuántica CNOT

Se prosigue a evaluar la compuerta, esta solo se aplica de dos bits cuánticos en adelante, en los estados $|00\rangle$, $|01\rangle$, $|10\rangle$ y $|11\rangle$.

$$CNOT|00\rangle = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} = |00\rangle$$

$$CNOT|01\rangle = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} = |01\rangle$$

$$CNOT|10\rangle = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} = |11\rangle$$

$$CNOT|11\rangle = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} = |10\rangle$$

Es interesante como se puede manejar dos bits cuánticos de esta manera y como el cambio de uno puede afectar o no el cambio de otro bit. se puede también plantear la definición de esta compuerta lógica como:

$$CNOT|xy\rangle, \quad x \rightarrow bit\ de\ control, \quad y \rightarrow bit\ controlado, \quad y = \bar{y} \leftrightarrow x = 1$$

Ahora se verá una nueva denotación que va a ser una introducción a los circuitos cuánticos. Con esta nueva denotación se va a estar aplicando la compuerta de Haddamard.

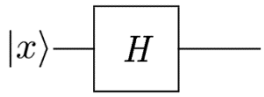


Ilustración 13. Compuerta de Haddamard aplicada a un solo bit.

Fuente de la imagen: Elaboración propia.

En la Ilustración 18 se puede apreciar un bit siendo afectado por una compuerta de Haddamard el cual puede tener los dos estados de $|0\rangle$ y $|1\rangle$, los cuales ya se resolvió cuáles serían las posibles resoluciones. Sin embargo, esta expresión se está aplicando para un solo

bit cuántico cuando la realidad no es esa. Entonces quiero plantear los 3 casos posibles de aplicar la compuerta de Haddamard ahora con dos bits cuánticos. Pero antes de plantear los casos, hay que definir un teorema que va a servir para darle una resolución a estos casos. Donde se va a plantear a A y B como operadores unitarios, y $|x\rangle$ y $|y\rangle$ como vectores en un espacio de Hilbert que representaran más adelante los bits cuánticos, de tal manera que:

$$A|x\rangle \otimes B|y\rangle = (A \otimes B)|xy\rangle$$

Ecuación 46. Teorema del producto de Tensor.

Caso 1:

Para el primer caso se va a plantear que la compuerta de Haddamard está afectando a los dos bits cuánticos de manera paralela.

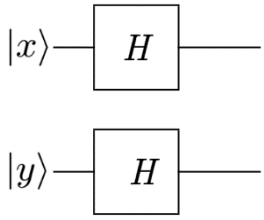


Ilustración 14. Compuerta de Haddamard aplicada a dos bits

Fuente de la imagen: Elaboración propia.

Aplicando la Formula 46 se puede deducir que:

$$H|x\rangle \otimes H|y\rangle = (H \otimes H)|xy\rangle$$

Entonces, ahora se resuelve para el producto tensor:

$$(H \otimes H) = \left(\frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix}\right) \otimes \left(\frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix}\right) = \frac{1}{2}\begin{pmatrix}1 & 1 & 1 & 1\\ 1 & -1 & 1 & -1\\ 1 & 1 & -1 & -1\\ 1 & -1 & -1 & 1\end{pmatrix}$$

Ahora esta matriz que también es unitaria estará afectando al sistema de dos bits cuánticos para cada estado posible, $|00\rangle$, $|01\rangle$, $|10\rangle$ y $|11\rangle$.

Caso 2:

Para el segundo caso se aplicará la compuerta de Haddamard solo para el primer bit cuántico.

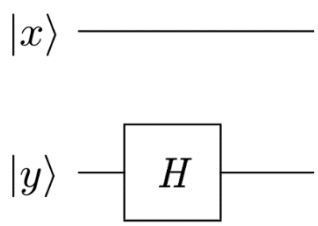


Ilustración 15. Compuerta de Haddamard aplicada al primer bit cuántico en un sistema de bit cuánticos.

Fuente de la imagen: Elaboración propia.

La pregunta ahorita es que operador está afectando al segundo bit si no hay nada en el circuito. El operador unitario que no afecta en nada a la matriz es la matriz identidad, que se denotara I. Entonces ahora se puede imaginar un bloque como el de H pero de I en el segundo bit cuántico de manera que se puede deducir con la Formula 46 que:

$$I|x\rangle \otimes H|y\rangle = (I \otimes H)|xy\rangle$$

Se resuelve ahora para el producto tensor:

$$(I \otimes H) = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \otimes \left(\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \right) = \begin{pmatrix} H & 0 \\ 0 & H \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & -1 \end{pmatrix}$$

Esta matriz, que también es unitaria es la que estará afectando a los bits cuánticos en estas situaciones para todos los estados posible, $|00\rangle$, $|01\rangle$, $|10\rangle$ y $|11\rangle$.

Caso 3:

Como es de esperarse el tercer caso sería cuando la compuerta de Haddamard afecta nada al segundo bit cuántico.

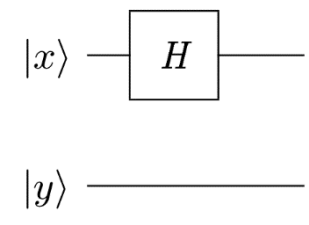


Ilustración 16. Compuerta de Haddamard aplicada al primer bit cuántico en un sistema de bit cuánticos.

Fuente de la imagen: Elaboración propia.

Entonces, volviendo a aplicar la Formula 46 para este caso y utilizando el mismo concepto del operador I en el primer bit. se puede encontrar que:

$$H|x\rangle \otimes I|y\rangle = (H \otimes I)|xy\rangle$$

se puede entonces resolver para el producto tensor:

$$(H \otimes I) = \left(\frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix}\right) \otimes \begin{pmatrix}1 & 0\\ 0 & 1\end{pmatrix} = \frac{1}{\sqrt{2}}\begin{pmatrix}I & I\\ I & -I\end{pmatrix} = \frac{1}{\sqrt{2}}\begin{pmatrix}1 & 0 & 1 & 0\\ 0 & 1 & 0 & 1\\ 1 & 0 & -1 & 0\\ 0 & 1 & 0 & -1\end{pmatrix}$$

Esta matriz, que también es un operador unitario, es la que va a estar afectando a todos los posibles estados de un sistema de dos bits cuánticos, $|00\rangle$, $|01\rangle$, $|10\rangle$ y $|11\rangle$.

Ahora, se planteará un cuarto escenario, solo que uno un poco más exagerado, pero a la misma vez aún más realista. Se plantea un caso donde la cantidad de bits cuánticos es n. La ecuación 45 es válida para cualquier cantidad de vectores, sin embargo, querer sacar el producto tensor de n cantidad de vectores no es muy práctico, como por ejemplo que $n = 100$ ó 1000 ó 10000. Para esto se va a definir una ecuación para la compuerta de Haddamard aplicada a n cantidad de bits cuánticos.

$$H_n|x\rangle = \frac{1}{2^{\frac{n}{2}}} \sum_{y=0}^{2^n-1} (-1)^{x \cdot y} |y\rangle$$

Ecuación 47. Ecuación de transformada de Haddamard para n cantidad de bits cuánticos

Ahora se analizará el comportamiento de la compuerta cuántica CNOT. Recordando entonces que esta compuerta cambia el estado de un bit cuántico, bit controlado, dependiendo del estado del otro bit cuántico, bit de control. De manera que el circuito cuántico de la compuerta CNOT se ve de la siguiente forma.

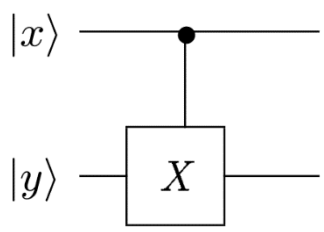


Ilustración 17. Compuerta CNOT

Fuente de la imagen: Elaboración propia.

Donde $|x\rangle$ es el bit cuántico de control y $|y\rangle$ es el bit cuántico controlado. Ahora se tendrán dos casos para esta compuerta.

Caso 1:

Para el primer caso se planteará un sistema de tres bits cuánticos, donde los primeros dos bits cuánticos son lo que están siendo afectados por la compuerta CNOT. De la siguiente manera:

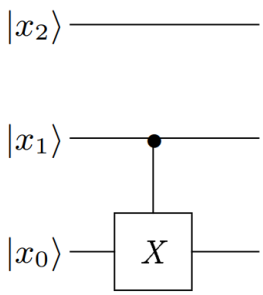


Ilustración 18. Aplicación de la compuerta CNOT afectando los primeros dos bits cuánticos en un sistema de tres bits cuánticos.

Fuente de la imagen: Elaboración propia.

Aplicando la Formula 46, la Formula 45 y recordando lo que se había planteado sobre el operador I, se puede observar lo siguiente:

$$I|x_2\rangle \otimes CNOT|x_1 x_0\rangle = (I \otimes CNOT)|x_2 x_1 x_0\rangle$$

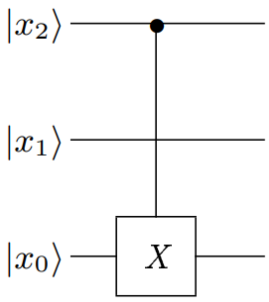
Donde entonces se puede resolver el producto tensor:

$$I \otimes CNOT = \begin{pmatrix} CNOT & 0 \\ 0 & CNOT \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

Lo más interesante de estos sistemas es ver la cantidad de información y estados que se pueden manejar en su sistema de tan solo tres bits cuánticos. Esta matriz, que también es un operador unitario, es la que estará afectando a todos los posibles estados del sistema de tres bits cuánticos, $|000\rangle$, $|001\rangle$, $|010\rangle$, $|011\rangle$, , $|100\rangle$, $|101\rangle$, $|110\rangle$ y $|111\rangle$.

Caso 2:

Para este segundo caso se planteará una situación, se tiene un sistema de tres bits cuánticos donde el primer bit cuántico es el bit controlado, el segundo es un bit cuántico que no está siendo afectado por la compuerta CNOT y el tercero es el bit cuántico de control.



Ecuación 48. Aplicación alterna de una compuerta CNOT en un sistema de tres bits cuánticos.

Fuente de la imagen: Elaboración propia.

En este caso el operador I esta de alguna manera interfiriendo con la compuerta CNOT entre los bits x_2 y x_0. Ahora lo lógico para este caso sería poder descomponer el CNOT en dos operadores unitarios de manera de que:

$$CNOT = A \otimes B \rightarrow A \otimes I \otimes B$$

Donde A y B son operadores unitarios, sin embargo, no hay tales operadores que ayuden a complacer la idea presentada arriba. Por lo tanto, se tiene que aplicar una solución más compleja y elegante para este sistema. Lo que se hará es plantear un operador unitario equivalente a una compuerta CNOT de dimensión $n\,x\,n$ y un vector que represente los posibles estados del sistema de tres bits cuánticos.

$$Ae_k = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1k} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2k} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nk} & \cdots & a_{nn} \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{pmatrix} = \begin{pmatrix} a_{1k} \\ a_{2k} \\ \vdots \\ a_{nk} \end{pmatrix}$$

Ecuación 49. Propiedad del CNOT con bits cuánticos de por medio de sus bits cuánticos de control y controlado.

Aquí A es el operador unitario equivalente de la compuerta CNOT y e_k el vector representante de los estados del sistema de n cantidad de bits cuánticos donde el 1 del vector está en la posición k.

3.1.11 Algoritmos y Circuitos Cuánticos

Los algoritmos cuánticos logran a través de varias características específicas sobre el mundo de la computación cuántica, como por ejemplo la superposición cuántica, para llegar de una entrada clásica, a través del estado de entrelazamiento, a una salida clásica de una manera más eficiente que el sistema clásico. Los algoritmos cuánticos son métodos utilizados en las redes y procesador cuánticos para resolver problemas algorítmicos. En un nivel más técnico y prosaico, un diseño de un algoritmo cuántico puede ser visto como un proceso de una eficiente descomposición de una compleja transformación unitaria a un producto de operaciones unitarias elementarías, ósea las compuertas cuánticas, generando simples cambios locales, (Gruska, 2000).

El esquema de un circuito que defina los pasos a seguir de las funciones cuánticas y las mediciones que afectan la forma de ver un algoritmo cuántico complejo puede resultar de gran utilidad, (Silva, 2018). Las compuertas cuánticas que ya se estuvieron analizando previamente pueden ser unidas de manera que armen un arreglo tipo red que abra la posibilidad de operar operaciones cuánticas, valga la redundancia, más complicadas que pueden realizar individualmente cada uno, (Deutsch, 1989).

Los algoritmos que se estarán viendo serán el algoritmo de Deutsch, algoritmo presentado para poder resolver un problema que se titula problema de Deutsch; se estará viendo el algoritmo de RSA, nombrado tras los tres científicos que reinventaron el algoritmo; luego se definirá la transformada de cuántica de Fourier, que servirá para plantear el siguiente algoritmo; el algoritmo de Shor, algoritmo que se presenta para poder enviar decodificar, o factorizar, un numero grande.

3.1.12 El Algoritmo de Deutsch

Primero se mirará el problema de Deutsch y su resolución. Ya se ha visto que la computadora cuántica universal puede perfectamente simular cualquier máquina de Turing y puede simularla con arbitraria precisión cualquier otra computadora cuántica o simulador. Ahora se debe comprobar que una computadora cuántica universal tiene la capacidad de

simular varios sistemas físicos, reales y teóricos, los cuales están más allá del alcance de una máquina universal de Turing, (Deutsch, 1985). El algoritmo presentado por David Deutsch para resolver los problemas que el planteaba es una ilustración de un modelo matemático. Entonces, gracias a este hombre, se tiene una excelente abstracción matemática que deja diseñar algoritmos cuánticos sin la necesidad de tener un extenso conocimiento de mecánica cuántica.

El planteamiento del problema de Deutsch empieza con una función que busca mapear un bit a un bit, y solo existen cuatro posibles funciones. Tal que:

$$f: \{0,1\} \rightarrow \{0,1\}$$

Ecuación 50. Definición de la función planteada para el problema de Deutsch.

$$f(x) = 0 \ \ (a), \qquad f(x) = 1 \ \ (b), \qquad f(x) = x \ \ (c), \qquad f(x) = \bar{x} \ \ (d)$$

Ecuación 51. Cuatro posibles funciones solución para el problema de Deutsch.

Donde, claramente, las funciona (a) y (b) de la Formula 51 son constantes y las funciones (c) y (d) son balanceadas.

Esta función se puede imaginar como una gran caja negra. La metáfora se implementa de manera que se puede ver resultados que presenta la caja negra pero no se puede ver como lo realiza, y solo se puede consultar como un oráculo. De manera que se le da una entrada 0 o 1 y devuelve una salida cualquiera. Deutsch decía que desafortunadamente solo le se puede hacer una consulta cada 24 horas al oráculo. La pregunta es, ¿qué tipo de función se encuentra en ese oráculo? Pero primero preguntémonos, ¿cuánto tiempo le toma a un sistema clásico? En el caso de un sistema clásico, parece que se tiene que hacer dos consultas de manera de saber que función es. Se piensa así, si se le da un valor de entrada al oráculo, se dice que $x = 1$, entonces el oráculo me devuelve $x = 0$, perfecto ahora se puede descartar las funciones (b) y (c) de la Formula 51. Sin embargo, se necesita una medida más para saber con certeza que función es, esto porque se sigue rebotando entre dos funciones, la (a) y la (d) de la Formula 51. Hasta otras 24 horas más que se vuelve a consultar al oráculo y volvió a decir que $x = 0$, se sabe con certeza que la función es la función (a) de la Fórmula 51 y por ende es una constante.

Para un sistema clásico no tiene esperanzas de poder hacer más rápido esta tarea. Pero en el caso de la computación cuántica se tiene un algoritmo que si tiene esperanza.

Como se ha aprendido anteriormente todas las operaciones cuánticas son ejecutadas por operadores unitarios y la información cuántica se presenta como vectores en un espacio de Hilbert. Dicho esto, se le puede dar una equivalencia cuántica a esta función que se llamará U_f. Se puede esperar entonces que esta equivalencia mapee un bit cuántico a un bit cuántico.

$$U_f|x\rangle|y\rangle \rightarrow |x\rangle|y \oplus f(x)\rangle$$

Ecuación 52. Definición de la equivalencia cuántica de la función f, oráculo.

Definido el oráculo, se dice que el bit cuántico $|y\rangle$ es igual a la base $|-\rangle$, de manera que:

$$U_f|x\rangle|-\rangle = \frac{1}{\sqrt{2}}|x\rangle(|0\rangle - |1\rangle) = \frac{1}{\sqrt{2}}(|x\rangle|0\rangle - |x\rangle|1\rangle) = \frac{1}{\sqrt{2}}(|x\rangle|f(x)\rangle - |x\rangle|1 \oplus f(x)\rangle)$$

Ecuación 53. Aplicación de la definición del oráculo utilizando la base |-⟩.

Ahora utilizando la ecuación 53 se va a poder ver que orientación toma el oráculo en base al valor de la función.

$$f(x) = 0, \qquad U_f|x\rangle|-\rangle = \frac{1}{\sqrt{2}}(|x\rangle|0\rangle - |x\rangle|1\rangle)$$

$$f(x) = 1, \qquad U_f|x\rangle|-\rangle = \frac{1}{\sqrt{2}}(|x\rangle|1\rangle - |x\rangle|0\rangle) = \frac{-1}{\sqrt{2}}(|x\rangle|0\rangle - |x\rangle|1\rangle)$$

En base a las expresiones arriba se puede concluir una ecuación para el oráculo, de manera que:

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

Reemplazando la base $|-\rangle$ por el estado del bit cuántico como tal, se puede definir entonces la siguiente fórmula para el oráculo:

$$U_f|x\rangle|y\rangle = (-1)^{f(x)}|x\rangle|y\rangle$$

Ecuación 54. Formula del oráculo.

Volviendo entonces a la ecuación 52, se va a asignarle las cuatro funciones al oráculo y ver que circuito cuántico equivale según la función asignada, planteando entonces cuatro casos.

Caso 1:

$$f(x) = 0, \qquad U_f|x\rangle|y\rangle \rightarrow |x\rangle|y \oplus f(x)\rangle = |x\rangle|y\rangle$$

Dado esto se puede concluir que el oráculo es igual a una matriz identidad ya que no altera los estados del sistema. De tal manera que el circuito cuántico del oráculo es el siguiente:

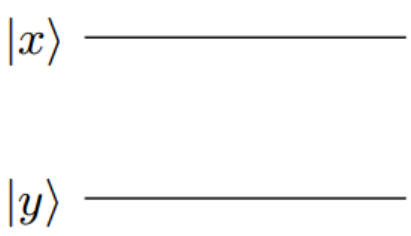


Ilustración 19. Esquema del circuito de Uf, cuando f(x)=0.

Fuente de la imagen: (Sysoev, 2019).

Caso 2:

$$f(x) = 1, \qquad U_f|x\rangle|y\rangle \rightarrow |x\rangle|y \oplus f(x)\rangle = |x\rangle|y \oplus 1\rangle$$

Dado esto se puede ver que el primer bit cuántico cambia de estado, entonces se puede concluir que el oráculo equivale a una matriz identidad en el segundo bit cuántico y una compuerta cuántica NOT en el primer bit. De tal manera que el circuito cuántico del oráculo es el siguiente:

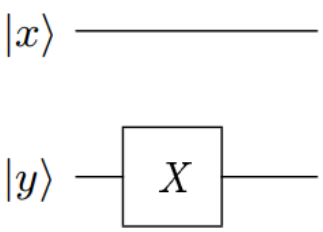


Ilustración 20. Esquema del circuito de Uf, cuando f(x)=1.

Fuente de la imagen: (Sysoev, 2019).

Caso 3:

$$f(x) = x, \quad x = 0 \rightarrow |x\rangle|y\rangle, \quad x = 1 \rightarrow |x\rangle|y \oplus 1\rangle$$

De tal manera que se mapea:

$$|00\rangle \rightarrow |00\rangle, \qquad |01\rangle \rightarrow |01\rangle, \qquad |10\rangle \rightarrow |11\rangle, \qquad |11\rangle \rightarrow |10\rangle$$

Esta es la definición de la compuerta cuántica CNOT, ya que el primer bit cuántico solo cambia si el segundo está en estado $|1\rangle$. Lo cual lleva a concluir que en este caso el oráculo tiene el esquema de una compuerta CNOT.

Caso 4:

$$f(x) = \bar{x}, \quad x = 0 \rightarrow |x\rangle|y \oplus 1\rangle, \quad x = 1 \rightarrow |x\rangle|y\rangle$$

De tal manera que se mapea

$$|00\rangle \rightarrow |01\rangle, \qquad |01\rangle \rightarrow |00\rangle, \qquad |10\rangle \rightarrow |10\rangle, \qquad |11\rangle \rightarrow |11\rangle$$

En este caso ocurre lo opuesto a la definición de una compuerta CNOT, en este caso el estado de bit cuántico controlado cambia si el estado del bit cuántico de control es $|0\rangle$. A esto se llama contraCNOT.

$$contraCNOT = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Ecuación 55. Compuerta cuántica contraCNOT.

Definido esto, se puede entonces concluir que el valor del oráculo es el de la compuerta cuántica contraCNOT. De tal manera que el circuito cuántico del oráculo es el siguiente:

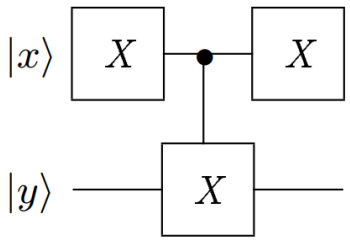


Ilustración 21. Compuerta cuántica contraCNOT/Esquema del circuito Uf, cuando f(x)= $\bar{x}$

Fuente de la imagen: (Sysoev, 2019).

Con todos los casos planteados para la resolución del problema planteado por Deutsch ahora se puede presentar el algoritmo de Deutsch:

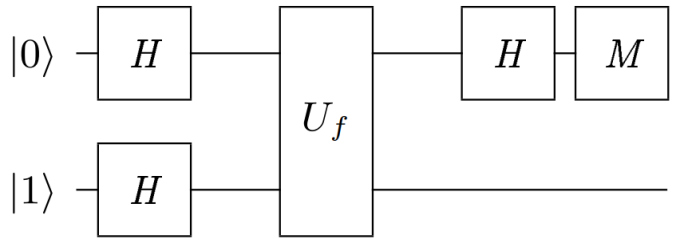


Ilustración 22. Algoritmo de Deutsch

Fuente de la imagen: (Sysoev, 2019).

Se puede entonces ver la acción que realiza el algoritmo de Deutsch para resolver el problema ya planteado:

$$|01\rangle \rightarrow H_2 \rightarrow \frac{1}{2}(|0\rangle + |1\rangle)(|0\rangle - |1\rangle) = \frac{1}{2}|0\rangle(|0\rangle - |1\rangle) + \frac{1}{2}|1\rangle(|0\rangle - |1\rangle) \rightarrow$$

$$U_f \rightarrow \frac{1}{2}(-1)^{f(0)}|0\rangle(|0\rangle - |1\rangle) + \frac{1}{2}(-1)^{f(1)}|1\rangle(|0\rangle - |1\rangle) = \frac{1}{\sqrt{2}}((-1)^{f(0)}|0\rangle + (-1)^{f(1)}|1\rangle)|-\rangle$$

Puestos en este punto se le dará dos formas cuando $f(x)$ es constante y cuando $f(x)$ es balanceada.

$$f(x) = const., \quad f(0) = f(1), \quad \rightarrow \frac{(-1)^{f(x)}}{\sqrt{2}}(|0\rangle + |1\rangle)|-\rangle = (-1)^{f(x)}|+\rangle|-\rangle \rightarrow$$

$$H \rightarrow |0\rangle|-\rangle$$

$$f(x) = bal., \ f(0) \neq f(1), \ \rightarrow \frac{1}{\sqrt{2}}\left((-1)^{f(0)}|0\rangle + (-1)^{f(1)}|1\rangle\right)|-\rangle = (-1)^{f(x)}(|0\rangle - |1\rangle)|-\rangle$$

$$H \rightarrow |1\rangle|-\rangle$$

Dado esto se puede concluir que debido a una entrada $|01\rangle$ si la salida del segundo bit cuántico es el estado $|0\rangle$ la función es constante, y si el estado es $|1\rangle$ la función es balanceada. Reduciendo el sistema a tan solo tener que consultar al oráculo una única vez.

Este mismo algoritmo se puede utilizar para resolver varios problemas como el problema de Deutsch-Jozsa, que trata de mapear n cantidad de bits cuánticos a un bit cuántico; el problema de Bernstein Vazirani que trata de mapear n cantidad de bits cuánticos a un bit cuántico, pero en vez de querer saber si la función es constante o balanceada, trata de encontrar la constante a de la función $f(x) = a \cdot x$; y también resuelve el problema de Simon que trata de mapear n cantidad de bits cuánticos a mapear n cantidad de bits

cuánticos, per este intenta encontrar la constante a de la función $y = a \oplus x$ donde $f(x) = f(y)$, sin embargo, debido a su conclusión este es el problema que más le cuesta resolver.

3.1.13 El Algoritmo de RSA

El algoritmo fue primero descubierto por Clifford Cocks en 1973, pero el algoritmo fue clasificado por el gobierno británico. Y no fue posible conseguir acceso al hasta en 1977 cuando Ron Rivest, Adi Shamir y Adleman lo reinventaron, y el algoritmo fue nombra tras las iniciales de los apellidos de cada uno de ellos.

El algoritmo de RSA, a groso modo, permite codificar un mensaje con una llave pública y una llave privada. Entonces el algoritmo funciona de la siguiente manera. Primero se va a tomar un número N que es producto de dos números primos, p y q, estos tres números siendo números grandes, por ejemplo, se puede decir que los números primos, p y q, son de diez mil dígitos cada uno lo que hacen a N un numero de veinte mil dígitos, y se quiere a partir de estos número codificar un mensaje utilizando otro conjunto de números. A partir de aquí se puede entonces describir las características de este algoritmo, que son las siguientes:

$$N = pq, \qquad p \neq q, \qquad \forall a < N, \qquad GCD(a, N) = 1$$

Ecuación 56. Propiedades para el algoritmo de RSA. Parte 1.

Ahora se tiene que recapitular algunas funciones y operadores, como sacar el modular de un numero con respecto a otro. La función que se tiene que analizar es la función $\Phi(N)$ que devuelve la cantidad de números que hay antes que el número N. Pero primero se tiene que plantear el teorema de Euler.

$$\forall a < N, \qquad GCD(a, N) = 1, \qquad a^{\Phi(N)} = 1(N)$$

Ecuación 57. Teorema de Euler.

Ahora para seguir planteando las características del algoritmo se tiene que evaluar N en la función de Euler.

$$\Phi(N) = \Phi(pq) = \Phi(p)\Phi(q) = (p-1)(q-1)$$

Ecuación 58. Propiedades para el algoritmo de RSA. Parte 2.

Ahora para seguir con el algoritmo se elige un numero e que cumpla con las propiedades planteadas en la Formula 56. Y se le llamará a (e, N) la llave pública de manera

que este numero e va ayudar a codificar el mensaje que se quiere enviar, por así decirlo. Ahora se elige un numero d, invertible en el campo $\mathbb{Z}_{\Phi(N)}$, que cumpla las siguientes características:

$$\exists d < N: ed = 1\big(\Phi(N)\big), \qquad \exists d, k: ed + \Phi(N)k = 1$$

Ecuación 59. Propiedades para el algoritmo de RSA. Parte 3.

Donde d y k son enteros positivos. Ahora se le llamará a (d, N) la llave privada para decodificar el mensaje. Entonces la acción del algoritmo va de manera que se elige un mensaje m que se puede codificar con la llave pública y decodificar con la llave privada, tal que:

$$codificar(m) = m^e(N) \rightarrow decodificar(codificar(m)) = (m^e)^d(N) = m^{ed}(N) = m^{\Phi(N)k+1}$$
$$= m^1 m^{\Phi(N)k} \rightarrow decodificar(codificar(m)) = m$$

Ecuación 60. Codificación y decodificación de un mensaje utilizando el algoritmo de RSA.

La idea de este algoritmo de RSA es que no se puede invertir fácilmente la separación de la exponenciación del módulo N.

3.1.14 La Transformada Cuántica de Fourier

En esta sección se va a estar planteada la transformada cuántica de Fourier, como se construye este operador y luego en el algoritmo de Shor se verá cómo se aplica de manera concreta.

Ahora se hablará sobre un operador que no se había mencionado, la transformada cuántica de Fourier. Esta transformada cuántica de Fourier es un operador unitario tal que:

$$N \coloneqq 2^n, \qquad QFT|x\rangle = \frac{\|x\|}{2^{n}/_2} \sum_{y=0}^{2^n-1} e^{2\pi i \frac{xy}{N}} |y\rangle$$

Ecuación 61. Transformada cuántica de Fourier.

Ahora, la transformada tiene que cumplir unas propiedades:

$$\left|\left|QFT|x\rangle\right|\right|^2 = \frac{\|x\|^2}{2^n}\left\langle \sum_{y=0}^{2^n-1} e^{2\pi i\frac{xy}{N}}|y\rangle \middle| \sum_{y=0}^{2^n-1} e^{2\pi i\frac{xy}{N}}|y\rangle \right\rangle = \frac{\|x\|^2}{2^n}\sum_{y=0}^{2^n-1} e^{(2\pi i-2\pi i)\frac{xy}{N}}\langle y|y\rangle$$

$$= \frac{\|x\|^2}{2^n}\sum_{y=0}^{2^n-1} 1 = \frac{\|x\|^2}{2^n}2^n = \|x\|^2$$

Por ende, la transformada cuántica de Fourier no afecta la norma del vector. Entonces, de ahora en adelante se va a obviar la norma $\|x\|$en la ecuación 61, ya que se está tratando con vectores unitarios. Ahora que pasa si se aplica la transformada cuántica de Fourier sobre el producto escalar de dos vectores ortogonales. Pero primero se tiene que definir la expresión para la suma de la progresión geométrica.

$$\sum_{y=0}^{2^n-1} a^y = \frac{a^{2^n}-1}{a-1}$$

Ecuación 62. Expresión para la suma de la progresión geométrica.

Por ejemplo, se tienen dos vectores unitarios, que se ven afectados por la transformada cuántica de Fourier de la siguiente manera:

$$\left|\left|x_1\right|\right| = \left|\left|x_2\right|\right| = 1, \qquad \langle x_1|x_2\rangle = 0$$

$$\langle QFT|x_1\rangle|QFT|x_2\rangle\rangle = \frac{1}{2^n}\left\langle \sum_{y=0}^{2^n-1} e^{2\pi i\frac{xy}{N}}|y_1\rangle \middle| \sum_{y=0}^{2^n-1} e^{2\pi i\frac{xy}{N}}|y_2\rangle \right\rangle$$

$$\frac{1}{2^n}\sum_{y_1=y_2}\left\langle e^{-2\pi i\frac{x_1y_1}{N}}|y_1\rangle \middle| e^{2\pi i\frac{x_2y_2}{N}}|y_2\rangle \right\rangle + \frac{1}{2^n}\sum_{y_1\neq y_2}\left\langle e^{-2\pi i\frac{x_1y_1}{N}}|y_1\rangle \middle| e^{2\pi i\frac{x_2y_2}{N}}|y_2\rangle \right\rangle =$$

$$\frac{1}{2^n}\sum_{y_1=y_2}\left\langle e^{-2\pi i\frac{x_1y_1}{N}}|y_1\rangle \middle| e^{2\pi i\frac{x_2y_2}{N}}|y_2\rangle \right\rangle = \frac{1}{2^n}\sum_{y=0}^{2^n-1} e^{\frac{2\pi i(x_2-x_1)y}{N}}$$

Ahora aplicando la Formula 62 se obtiene que:

$$\frac{e^{2\pi i(x_2-x_1)}-1}{e^{\frac{2\pi i(x_2-x_1)}{N}}-1} = \frac{1-1}{e^{\frac{2\pi i(x_2-x_1)}{N}}-1} = 0$$

Se puede observar entonces que la transformada cuántica de Fourier conserva los ángulos, mapeando un base ortogonal a un base ortogonal. Entonces pero este motivo y la conservación de la norma ya explicada se puede definir a la transformada cuántica de Fourier como un operador unitario.

Luego de ejecutar la sumatoria de la transformada cuántica de Fourier se puede describir como:

$$QFT|x\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle + e^{2\pi i\frac{x_0}{2}}|1\rangle\right) \otimes \frac{1}{\sqrt{2}}\left(|0\rangle + e^{2\pi i\left(\frac{x_0}{4}+\frac{x_1}{2}\right)}|1\rangle\right) \otimes \ldots \otimes \frac{1}{\sqrt{2}}\left(|0\rangle + e^{2\pi i\left(\frac{x_0}{2^n}+\frac{x_1}{2^{n-1}}+\cdots+\frac{x_{n-1}}{2}\right)}|1\rangle\right)$$

Tal parece que la transformada cuántica puede ser descompuesta a productos tensores de operadores de sistemas de un solo bit cuántico. Permitiendo entonces poder plantear la siguiente compuerta cuántica:

$$R_k = \begin{pmatrix} 1 & 0 \\ 0 & e^{2\pi i\frac{1}{2^k}} \end{pmatrix}$$

Ecuación 63. Compuerta cuántica descompuesta de la transformada cuántica de Fourier para sistemas de un bit cuántico.

Utilizando entonces esta nueva compuerta y la compuerta de Haddamard se puede llegar a una equivalencia para la transformada cuántica de Fourier.

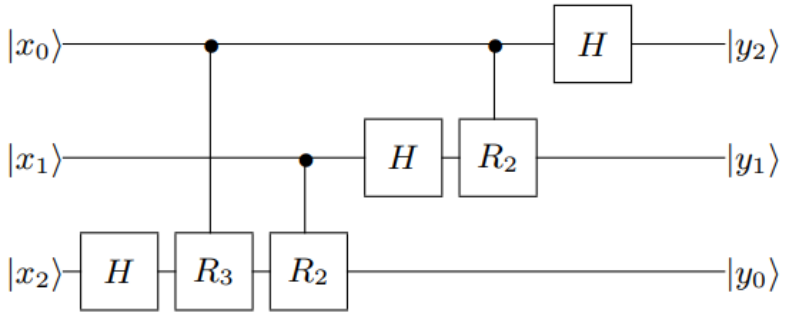


Ilustración 23. Equivalencia de la transformada cuántica de Fourier.

Fuente de la imagen: (Sysoev, 2019).

Si se generaliza la Ilustración 28 para una n cantidad de bits cuánticos, la complejidad de las implementaciones de la transformada cuántica de Fourier se convierte en la siguiente expresión.

$$\sum_{j=1}^{n} j \approx O(n^2)$$

Ecuación 64. Complejidad de implementación de la transformada cuántica de Fourier.

3.1.15 El Algoritmo de Shor

El algoritmo de RSA es un algoritmo polinómico en el tiempo para logaritmos discretos y factorización de una computadora cuántica. Está siendo una los resultados más significantes de la computación cuántica. El algoritmo fue inventado por Peter Shor en 1994 y logra en un tiempo $logN$ realizar la factorización del numero N.

La relación que tiene este algoritmo con el algoritmo de RSA es que este algoritmo es probablemente el único método posible para decodificar el mensaje encriptado con la llave publica y decodificable con la llave privada. El algoritmo se implementa tal que su esquema del circuito se ve de la siguiente manera:

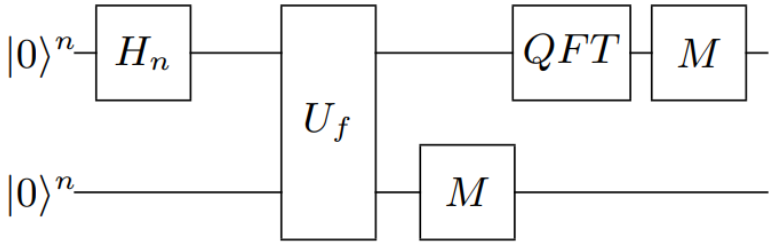


Ilustración 24. Algoritmo de Shor.

Fuente de la imagen: (Sysoev, 2019).

Antes de poder entender la manera en la que funciona el algoritmo se tiene que entender unas características que lo rodean.

$$f: \{0,1\}^n \to \{0,1\}^n, \qquad r: \forall x, \qquad f(x+r) = f(x), \qquad f_a = a^x(pq), \qquad N = 2^n$$

Ecuación 65. Propiedades para el algoritmo de Shor.

Sabiendo las propiedades necesarias que rigen el algoritmo, se puede ver que el algoritmo función de tal manera que:

$$|0\rangle^n|0\rangle^n \to H_n \to \frac{1}{2^{n}/_2}\sum_{x=0}^{2^n-1}|x\rangle|0\rangle^n \to U_f \to \frac{1}{2^{n}/_2}\sum_{x=0}^{2^n-1}|x\rangle|f(x)\rangle \to$$

$$Measure\ |y\rangle \to \frac{1}{\sqrt{A}}\sum_{j=0}^{A-1}|x_0 + jr\rangle\big|f(x_{0)}\big\rangle$$

Donde;

$$A \approx \frac{N}{r}, \qquad N - r \leq Ar \leq N + r,$$

Sabiendo el valor de A y que nada más interesa medir el segundo bit cuántico. De tal manera que la acción del algoritmo continua con la aplicación de la transformada cuántica de Fourier:

$$\frac{1}{\sqrt{A}} \sum_{j=0}^{A-1} |x_0 + jr\rangle \rightarrow QFT \rightarrow \frac{1}{\sqrt{AN}} \sum_{j=0}^{A-1} \sum_{y=0}^{2^n-1} e^{2\pi i \frac{x_0 y}{N}} e^{2\pi i \frac{jry}{N}} |y\rangle =$$

$$= \frac{1}{\sqrt{AN}} \sum_{y=0}^{2^n-1} e^{2\pi i \frac{x_0 y}{N}} \sum_{j=0}^{A-1} e^{2\pi i \frac{jry}{N}} |y\rangle$$

Debido a que;

$$e^{2\pi\gamma} = 1$$

Donde γ es cualquier entero complejo. Sabiendo esto se prosigue a medir la posibilidad de que cualquier $|y\rangle$ particular resultado de la medición, utilizando la Formula 20, se obtiene que:

$$P(|y\rangle) = \left(\frac{1}{\sqrt{AN}} \sum_{y=0}^{2^n-1} e^{2\pi i \frac{x_0 y}{N}} \sum_{j=0}^{A-1} e^{2\pi i \frac{jry}{N}} \right) = \frac{1}{AN} \left| \sum_{j=0}^{A-1} e^{2\pi i \frac{jry}{N}} \right|^2$$

Ecuación 66. Probabilidad del estado $|y\rangle$ después de la segunda medición en la resolución del algoritmo de Shor.

Pudiendo denotar;

$$\theta_y = 2\pi \frac{ry(N)}{N}$$

Ecuación 67. Fase de ángulo de la transformada cuántica de Fourier.

Se puede reescribir la expresión del coeficiente del estado $|y\rangle$, aplicando la Formula 67 y la Formula 62, como:

$$\sum_{j=0}^{A-1} e^{2\pi i \frac{jry}{N}} = \sum_{j=0}^{A-1} e^{\theta_y ij} = \frac{e^{A\theta_y i} - 1}{e^{\theta_y i} - 1}$$

Ecuación 68. Expresión del estado $|y\rangle$ después de medición del algoritmo expresada para el ángulo de desfase de la transformada de Fourier.

3.2 FPGA

En esta parte del capítulo se explicará lo que se tiene que saber sobre la FPGA, su origen, el lenguaje de programación que se utiliza, la placa de desarrollo que se va a implementar para la investigación y del software que se utilizara para realizar el código.

Un FPGA es un circuito integrado diseñado para ser configurado por el usuario o un diseñador por manufactura. Un FPGA contiene arreglos de bloques lógicos programable, y una jerarquía de conexiones internas reconfigurables que permiten a los bloques estar conectados juntos, como muchas otras configuraciones lógicas puede estar conectado internamente en diferentes configuraciones. Los bloques lógicos pueden ser configurados para realizar complejas funciones combinatorias, o simplemente simples compuertas lógicas como AND y XOR. En la mayoría de las FPGAs, los bloques lógicos también incluyen elementos de memoria, que pueden ser simples flip-flops o bloques de memoria más completos, (Barkalov y Titarenko, 2008)

La industria de los FPGAs broto de las memorias programables de solo lectura, o PROMs, por sus siglas en ingles programable read-only memory), y de los dispositivos con lógica programmable, o PLDs, por sus siglas en inglés programmable logic device. Los PROMs y PLDs tenían la opción de ser programados en lotes en una fábrica o en el campo. Sin embargo, estos dispositivos eran conectados con cables entre las compuertas lógicas, («WayBackMachine», 2015).

3.2.1 MIMAS V2 SPARTAN 6 FPGA DEVELOPMENT BOARD

Mimas V2 es una tarjeta de desarrollo de FPGA de bajo coste con Xilinx Spartan-6 FPGA. MIMAS V2 está especialmente diseñada para experimentar y diseñar sistemas de aprendizaje con FPGAs. Esta placa de desarrollo cuenta con Spartan XC6SLX9 CSG324 FPGA con 512Mb DDR SDRAM. La interfaz USB 2.0 proporciona una descarga de configuración rápida y fácil al flash SPI incorporada, (*Numato Lab*, s. f.).

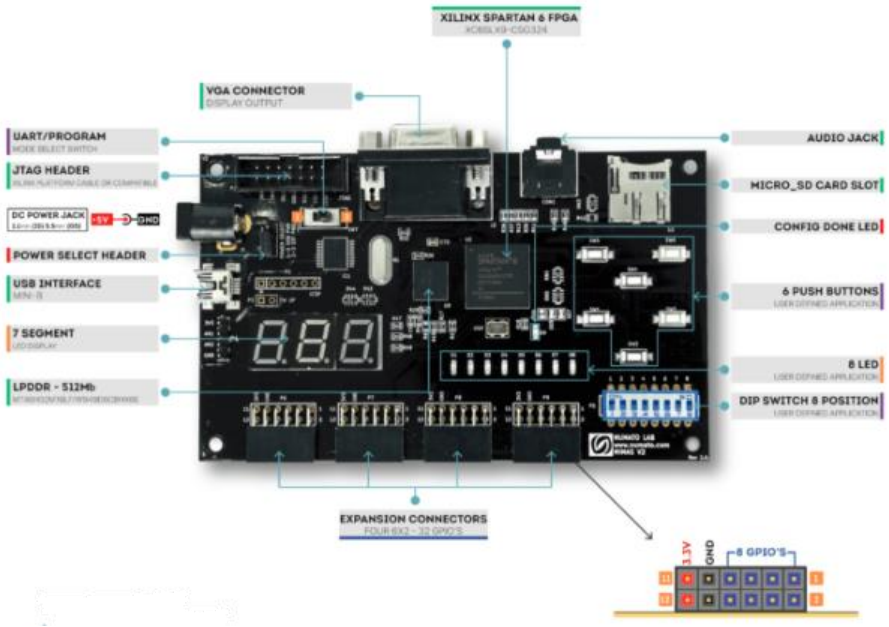
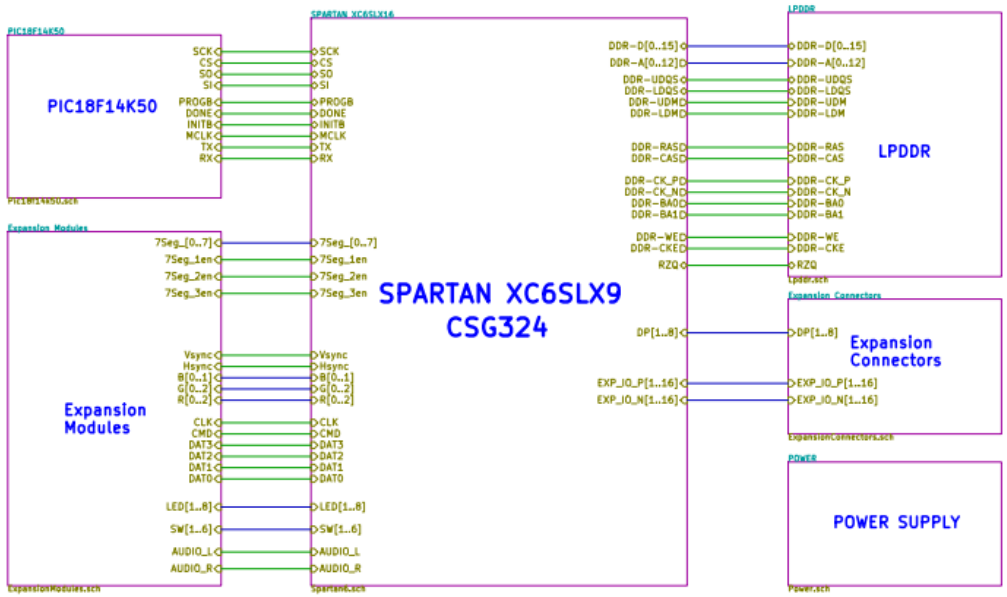


Ilustración 25. Placa de desarrollo Mimas V2 con sus especificaciones físicas.

Fuente de la imagen: (*Numato Lab*, s. f.)

La placa de desarrollo está conformada por un FPGA: SPARTAN XC6SLX9 CSG324, una memoria DDR: 66MHz 512Mb LPDDR, una memoria flash: 16 Mb SPI flash memory (M25P16), configuración del FPGA vía JTAG y USB, conector VGA, jack de estéreo, un adaptador de micro SD, 3 displays de siete segmentos, 32 entradas y salidas, y 4 expansores de 6 x 2.



Ecuación 69. Esquemático de la placa Mima V2.

Fuente de la Imagen: (*Numato Lab*, s. f.)

3.2.2 XILINX ISE DESIGN SUITE 14.7 WEBPACK

El software de diseño ISE WebPACK es la única solución de diseño FPGA gratuita de la industria para Linux, Windows XP y Windows 7. ISE WebPACK es la solución descargable ideal para el diseño de FPGA y CPLD que ofrece síntesis y simulación HDL, implementación, ajuste de dispositivos y programación JTAG. ISE WebPACK ofrece un flujo de diseño completo, de principio a fin, proporcionando un acceso instantáneo a las características y funcionalidad de ISE sin coste alguno. Xilinx ha creado una solución que permite una productividad conveniente al proporcionar una solución de diseño que siempre está actualizada con descarga sin errores e instalación de un solo archivo.

3.3 METODOLOGÍA

3.3.1 MODELO INCREMENTAL

Para poder tener un camino que seguir en un proyecto de investigación es importante poder definir una metodología que se adapte al proyecto. En el caso de este proyecto de investigación se planteó como metodología de investigación el modelo incremental. Fue propuesta por Harlan D. Mills en 1980, el propósito es poder definir un ciclo de procesos en forma de cascada para poder someter el sistema, o software, en diferentes tareas o bajo diferentes objetivos que según se cumplan estos se va aumentando la complejidad de estos objetivos.

Una serie de actividades se definen de manera que describan un proceso de software desde la definición del sistema hasta el soporte operacional, (Mills, 1980). Estas actividades que componen el proceso del software son mejor definidas en términos de componentes de trabajo que identifiquen las tareas a ser realizadas, en la siguiente ilustración.

Actividad	Componentes de Trabajo
Definición del sistema	Defeniciones requeridas para el software Descripción del sistema del software Planeación del desarrollo del software
Diseño del software	Diseño funcional Diseño del programa Pruebas del diseño Herrmaientas del software Evaluación del diseño
Desarrollo del software	Desarrollo del modulo Pruebas del desarrollo
Pruebas del sistema del software	Procedimientos de las pruebas del sistema de software Integración del software y pruebas
Sistemas y pruebas de aceptación	Pruebas de software de soporte Pruebas de aceptación de soporte
Soporte Operacional	Operación de sistema de soporte Entrenamiento Soporte al despliegue del sitio

Tabla 6. Ciclo de vida de las actividades de un software.

Fuente de la imagen: (Mills, 1980).

De manera entonces que estas actividades luego de cada ciclo se repiten bajo un objetivo más complejo, y con más funcionalidades y características.

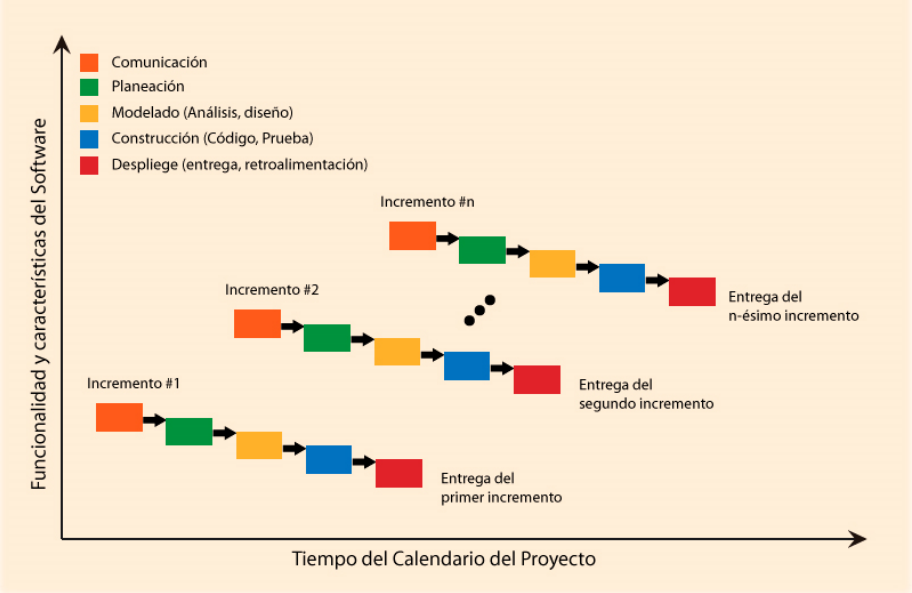


Ilustración 26. Modelo Incremental.

Fuente de la imagen: (Ortiz, 2017)

CAPÍTULO 4. METODOLOGÍA

4.1 ENFOQUE

Teniendo como finalidad poder ver la capacidad de la FPGA al momento de poder implementar el algoritmo de Deutsch en dos distintos problemas. Dicho esto, el proyecto se le dio un enfoque cuantitativo. Esto debido a que el análisis de la emulación de la tecnología cuántica se da bajo distintas características y capacidades de procesamiento.

Dicho lo anterior, se puede decir que la investigación es de diseño experimental, poniendo así, las diferentes variables bajo diferentes parámetros y observando la diferencia que estas tienen bajo estos distintos parámetros.

4.2 VARIABLES DE INVESTIGACIÓN

Si es cierto que hay muchos factores que pueden perjudicar los casos a los que se van a someter los sistemas considerados para la investigación, las siguientes variables son la denotadas como más cruciales para la conclusión de la tesis. De tal manera entonces que, las variables fueron clasificadas de la siguiente manera.

4.2.1 Variables Dependientes

Como variable dependiente se plantearán los bits para la medición de la funcionalidad de los circuitos digitales que se estarán realizando para la simulación y emulación de los algoritmos y compuertas cuánticas.

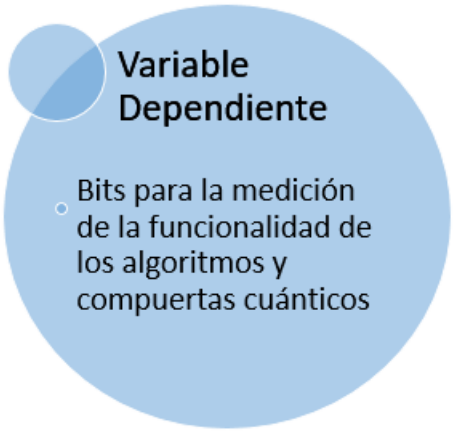


Ilustración 27. Variables dependientes.

Fuente de la imagen: Elaboración propia.

4.2.2 Variables Independientes

Este tipo de variable va a variar en los ciclos de la investigación. En el primer ciclo que se estará trabajando con las compuertas cuánticas no se van a plantear variables independientes ya que la funcionalidad va a depender enteramente de los bits que se les asignaran a los parámetros a medir. Sin embargo, las mismas compuertas cuánticos serán factores importantes al momento de la funcionalidad entera de los algoritmos cuánticos. Por lo que estos circuitos equivalentes serán las variables independientes de la investigación.

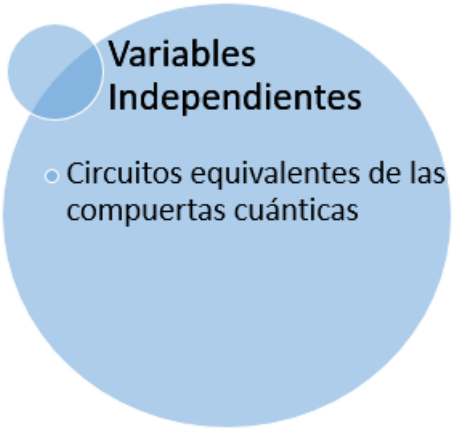


Ilustración 28. Variables Independientes.

Fuente de la imagen: Elaboración propia.

4.3 HIPÓTESIS DE INVESTIGACIÓN

Por medio de una hipótesis de investigación, se puede entonces declarar una suposición o especular los resultados que posiblemente puede llegar a brindar la investigación. De tal manera que, se declaró la siguiente hipótesis de investigación.

H_i: Los diseños realizados para las equivalencias cuánticas se comportarán de tal forma como lo hacen los conceptos matemáticos de estas, y la implementación en la FPGA será exitosa.

4.4 TÉCNICAS E INSTRUMENTOS APLICADOS

Para la presente investigación inicialmente se planteó poder investigar sobre la computación cuántica y poder ver la mejoría que esta tecnología promete. Se investigó, a través de diferentes revistas académicas, tesis universitarias y libros, que dispositivo o tecnología se acerca o ayudaría a poder hacer un análisis, por medio de una emulación, de la tecnología cuántica. Por medio de esta investigación se llegó a la definición del uso del FPGA para la emulación.

Existen varias herramientas ingenieriles que permiten poder configurar los FPGA y poder emular algoritmos cuánticos. Para la presente investigación se estarán utilizando las siguientes herramientas.

1. Xilinx ISE Design Suite 14.7 WebPACK, un software que permite programar en VHDL o Verilog.
2. Digital: Es un software que permite el diseño por bloques de los circuitos digitales que permite exportar código a VHDL o Verilog.
3. Karnaugh Map Minimizer: Es un software que permite resolver un sistema digital utilizando el sistema del mapa de Karnaugh, con una limitante de ocho variables máximo.

4.5 Materiales

En este caso, sabiendo que se va a utilizar una FPGA, la lista de materiales será corta, ya que básicamente se basa en un solo material que es: Mimas V2 Spartan 6 FPGA Development Board.

La placa elegida, que fue descrita en la sección 3.2.1 de la presente tesis, fue considerada mayormente por su valor económico. Ya que su valor es muy accesible y brinda una gran variedad de extensiones que se utilizarán, como ser, dip switches, botoneras, expansores y luces LEDs, además de claro el FPGA como tal que es un Spartan 6.

4.6 Metodología de Estudio

Siendo utilizada una metodología incremental de manera de poder ir aumentando la dificultad de computación de los sistemas planteados y poder ir analizando la mejoría que tienen estas emulaciones con FPGA para algoritmos cuánticos con respecto a la emulación que se puede hacer en un modelo clásico de computación. De manera que las etapas serán definidas en base a los sistemas o problemas a los que se someterá la investigación. Definiendo entonces las siguientes etapas.

Etapa I: Compuertas Cuánticas.

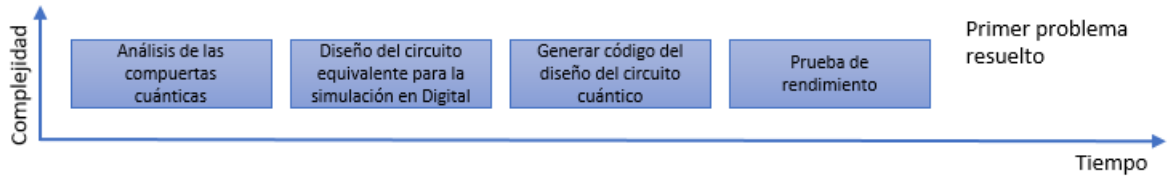


Ilustración 29. Primer ciclo de la metodología incremental.

Fuente de la imagen: Elaboración propia.

Etapa I.I: Análisis de las compuertas cuánticas.

En esta primera parte de la primera etapa se analizará el concepto matemático de las cuatro compuertas cuánticas y sintetizando sus funciones. De esta forma, poder generar un concepto de cambios, ya sean de, variables u operaciones aritméticas, que sean necesarios para poder diseñar las equivalencias de las compuertas cuánticas.

Etapa I.II: Diseño del circuito cuántico equivalente para la simulación en Digital.

En esta parte se estará aplicando el análisis de la etapa anterior de manera de poder diseñar un circuito equivalente en base a las representaciones que se estará asignando de las cuatro compuertas cuánticas necesarias para poder aplicar el algoritmo de Deutsch.

Etapa I.III: Generar código.

En esta etapa, se exportarán los circuitos diseñados en Digital a Verilog de manera de poder correr el código en ISE Design Suite. De manera de poder generar el archivo bin necesario para poder reprogramar la FPGA.

Etapa I.IV: Poner a prueba el circuito.

En esta etapa final del ciclo, se medirá la funcionalidad de los circuitos diseñados para la equivalencia digital con respecto a la cuántica comparándolo con el concepto matemático expuesto para las compuertas cuánticas.

ETAPA II: ALGORITMO DE DEUTSCH: APLICADO AL PROBLEMA DE DEUTSCH.

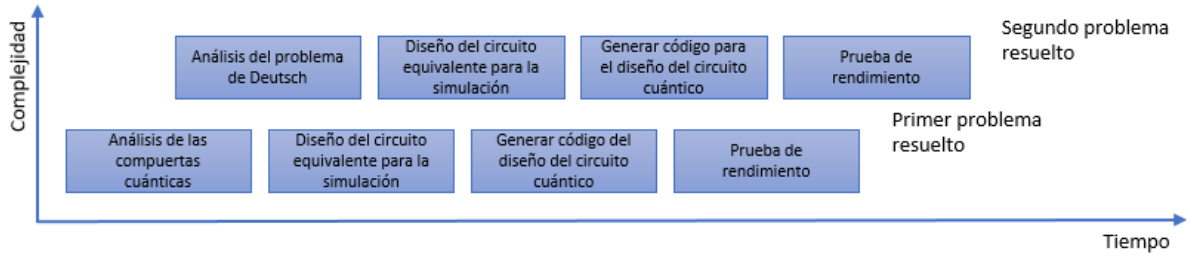


Ilustración 30. Segundo ciclo de la metodología incremental.

Fuente de la imagen: Elaboración propia.

Etapa II.I: Análisis del problema de Deutsch.

En esta primera parte del segundo ciclo, se analizará el algoritmo de Deutsch de manera de poder saber si se tienen las herramientas necesarias para poder realizar un diseño equivalente en el software Digital. De manera de poder solucionar el problema planteado por Deutsch.

Etapa II.II: Diseño del circuito cuántico del algoritmo de Deutsch en base al problema de Deutsch.

En esta etapa se estará diseñando el circuito equivalente necesario para la representación del algoritmo de Deutsch aplicado al problema de Deutsch. Utilizando las herramientas acumuladas en el primer ciclo de manera de conseguir un circuito equivalente eficaz y lo más sintetizado posible.

Etapa II.III: Generar código.

En esta etapa se exportará el diseño realizado en el software Digital a código Verilog de manera de poder aplicar este código en el software ISE Design Suite. De manera de poder entonces generar el archivo bin necesario para reprogramar la FPGA.

Etapa II.IV: Poner a prueba el circuito.

En esta etapa final del ciclo, se estará midiendo la funcionalidad de los diseños realizados para la equivalencia lógica del algoritmo de Deutsch y de su implementación en la FPGA.

ETAPA III: ALGORITMO DE DEUTSCH: APLICADO AL PROBLEMA DE DEUTSCH-JOZSA.

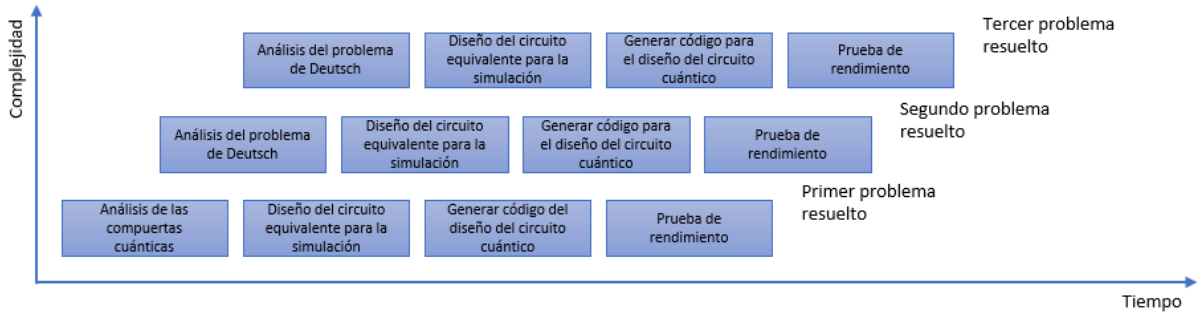


Ilustración 31. Tercer ciclo de la metodología incremental.

Fuente de la imagen: Elaboración propia.

Etapa III.I: Análisis del problema de Deutsch-Jozsa.

En este ciclo se estará aplicando el algoritmo de Deutsch para resolver el problema de Deutsch-Jozsa. Se analizará la capacidad que puede tener el FPGA de manera de saber cuántos bits cuánticos se estarán agregando. Y poder observar si se tienen las herramientas necesarias para el diseño y simulación en el software Digital

Etapa III.II: Diseño del circuito cuántico del algoritmo de Deutsch en base al problema de Deutsch-Jozsa.

Aquí se replanteará el algoritmo de Deutsch en base al problema de Deutsch-Jozsa como un circuito cuántico. A este circuito cuántico se le dará una equivalencia en base a un sistema lógico. De manera de diseñar el circuito como un circuito digital de manera de saber las compuertas lógicas que lo estarán efectuando, claro está que habrá varios de estos circuitos trabajando paralelamente.

Etapa III.III: Generar código.

En esta etapa se programará el código necesario para cumplir el flujo propuesto para poder dar una equivalencia digital paralela del circuito cuántico necesario para poder procesar el algoritmo de Deutsch en base al problema de Deutsch-Jozsa.

Etapa III.IV: Poner a prueba el circuito.

En esta etapa final del ciclo, se estará midiendo la funcionalidad que tiene el circuito lógico digital diseñado para la equivalencia del circuito cuántico necesario para ejecutar el algoritmo de Deutsch aplicado al problema de Deutsch-Jozsa.

4.7 Cronograma de Actividades

Para la presente investigación se consideraron diferentes actividades para la realización más efectiva del presente proyecto. Por lo que se seccionaron las actividades a realizar de manera de poder cubrir todos los objetivos en el tiempo establecido. Por lo que primero se trabajó con el planteamiento del problema, de manera de poder plantear las preguntas que rodean la investigación. Luego se investigó sobre la teoría de sustento necesaria para la realización y comprensión del proyecto. Posterior a eso se planteó la metodología necesaria para la realización eficaz del proyecto. Planteado todo esto se comenzó a trabajar con el proyecto de manera de conseguir los análisis y resultados para poder dar las conclusiones a las preguntas planteadas para la investigación.

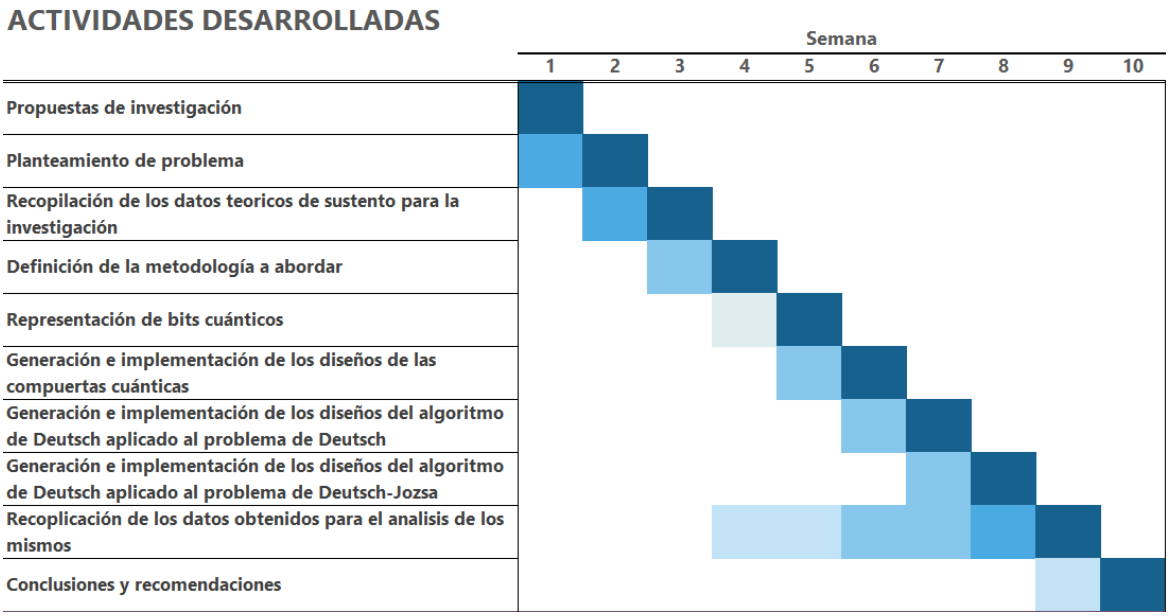


Ilustración 32. Cronograma de Actividades.

Fuente de la imagen: Elaboración propia.

Para la mejor comprensión de la tabla, se define el tono del color como el porcentaje de completación por cada semana.

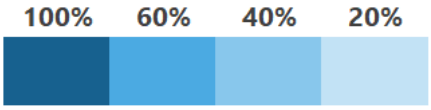


Ilustración 33. Porcentaje de completación de actividad en base al tono de color.

Fuente de la imagen: Elaboración propia.

CAPÍTULO 5. ANÁLISIS Y RESULTADOS

En este capítulo se estará demostrando los análisis necesarios para la generación de los diseños necesarios para la simulación e implementación de los algoritmos cuánticos. Para la presente tesis se eligieron el algoritmo de Deutsch aplicado a dos distintos problemas, el problema de Deutsch y el problema de Deutsch-Jozsa, este último aplicado a 3 bits cuánticos. Sin embargo, por limitantes como ser la gamma de programación y el hardware utilizado solo se podrá dejar como base algunas compuertas cuánticas y como sugerencia la implementación de más bits cuánticos y la emulación de algoritmos más complejos.

5.1 Representación de los Bits Cuánticos

5.1.1 Un Bit Cuántico

El primer paso para poder realizar este proyecto es poner un compromiso para saber cómo poder representar, el núcleo de todo este proyecto, que es el bit cuántico. Como se puede recordar de la fórmula para un bit cuántico el estado del bit depende de sus dos bases que son el estado |0⟩ y el estado |1⟩. También se tiene que considerar que estos bits están en una superposición, entonces la probabilidad de que estos estados ocurran depende de un escalar.

Sabiendo lo anterior se definirá que para un solo bit cuántico se utilizarán 18 bits lógicos clásicos. Asignando a cada escalar 6-bits, al signo de cada escalar 1-bit y a los estados 2-bits. Las divisiones de los bits se ven mejor ilustrados en la siguiente tabla.

Bit Cuántico	**Bit Clásico**
|ϕ>=α|0>+β|0>	18-bit
$\|\varphi\rangle = a_{17}a_{16}a_{15} \dots a_2a_1a_0$	
División de los bits clasicos	
Sección del bit cuanticó	**Bits clasicos asignados**
α	6-bits
$\alpha = a_{16}a_{15}a_{14}a_{13}a_{12}a_{11}$	
Signo de α	1-bits
Signo de $\alpha = a_{17}$	
|0>	2-bits
$\alpha\|0\rangle = a_{17}a_{16}a_{15}a_{14}a_{13}a_{12}a_{11}a_{10}a_9$	
β	6-bits
$\beta = a_7a_6a_5a_4a_3a_2$	
Signo de β	1-bit
Signo de $\beta = a_{18}$	
|1>	2-bits
$\beta\|1\rangle = a_8a_7a_6a_5a_4a_3a_2a_1a_0$	

Tabla 7. Repartición de los bits clásicos para la representación de un bit cuántico.

Fuente de la imagen: Elaboración propia.

Para comprender bien la ilustración 40 se puede pensar en cada uno de los a_n como un digito del conjunto de bits, no confundir con una multiplicación. Sabiendo entonces que los escalares α y β son representados por 6 bits es importante saber los valores que se estarán manejando con estos. Sabiendo la ecuación 18 se puede deducir que los escalares α

y β nunca van a ser mayores a 1, y que cuando uno de ellos es 1 el otro es 0. Entonces para saber que valores aproximados que se manejarán es importante asignarle a cada bit un valor.

Bit	Valor asignado
$a_{16,7}$	2^0
$a_{15,6}$	2^{-1}
$a_{14,5}$	2^{-2}
$a_{13,4}$	2^{-3}
$a_{13,3}$	2^{-4}
$a_{11,2}$	2^{-5}

Tabla 8. Valores asignados a cada bit para los escalares α y β.

Fuente de la imagen: Elaboración propia.

Es importante saber los valores que se pueden estar utilizando para aproximar las constantes que se estarán utilizando. Aplicando, entonces, la Ilustración 41 para conocer los valores que se puede asignar a los escalares.

$a_{16,7}$	$a_{15,6}$	$a_{14,5}$	$a_{13,4}$	$a_{12,3}$	$a_{11,2}$	Valor	$a_{16,7}$	$a_{15,6}$	$a_{14,5}$	$a_{13,4}$	$a_{12,3}$	$a_{11,2}$	Valor
0	0	0	0	0	0	0	0	1	0	0	0	1	0.5313
0	0	0	0	0	1	0.0313	0	1	0	0	1	0	0.5625
0	0	0	0	1	0	0.0625	0	1	0	0	1	1	0.5938
0	0	0	0	1	1	0.0938	0	1	0	1	0	0	0.625
0	0	0	1	0	0	0.125	0	1	0	1	0	1	0.6563
0	0	0	1	0	1	0.1563	0	1	0	1	1	0	0.6875
0	0	0	1	1	0	0.1875	0	1	0	1	1	1	0.7188
0	0	0	1	1	1	0.2188	0	1	1	0	0	0	0.75
0	0	1	0	0	0	0.25	0	1	1	0	0	1	0.7813
0	0	1	0	0	1	0.2813	0	1	1	0	1	0	0.8125
0	0	1	0	1	0	0.3125	0	1	1	0	1	1	0.8438
0	0	1	0	1	1	0.3438	0	1	1	1	0	0	0.875
0	0	1	1	0	0	0.375	0	1	1	1	0	1	0.9063
0	0	1	1	0	1	0.4063	0	1	1	1	1	0	0.9375
0	0	1	1	1	0	0.4375	0	1	1	1	1	1	0.9688
0	0	1	1	1	1	0.4688	1	x	x	x	x	x	1
0	1	0	0	0	0	0.5							

Tabla 9. Valores según la tabla de verdad de los bits asignados a los escalares

Fuente de la imagen: Elaboración propia.

Pero ahora, ¿qué ocurriría si se estuviera trabajando con más de un bit cuántico?, lo que es esencial preguntarse ya que todos los sistemas funcionan con más de un bit cuántico. Para lo que se puede calcular la cantidad de bits clásicos necesarios para representar una cierta cantidad de bits cuánticos.

La fórmula se construyó en base a una suma por sección del bit cuántico. Cuando se hace referencia a las secciones de un bit cuántico se refiere a las partes que lo conforman, como ser los escalares, que se puede dividir en magnitud y signo, y los estados. Entonces si se quiere encontrar la cantidad de bits clásicos necesarios para la magnitud de los escalares se debe multiplicar la cantidad de escalares por la cantidad de bits asignados a estos valores, que son 6 bits. La cantidad de escalares es dependiente de la dimensión del vector. En este caso, son el mismo valor. Ya que cada escalar representa la probabilidad de que cada base

ocurra. Dicho esto, se puede representar la fórmula de la siguiente manera, donde se va a representar a esta cantidad de bits clásicos como a.

$$a = 7 * 2^n$$

Ecuación 70. Cantidad de bits clásicos para representar las magnitudes de los escalares para una n cantidad de bits cuánticos.

Luego, para obtener la cantidad de bits clásicos necesarios para definir el signo de los escalares, al igual que la sección de la magnitud, se ocupa multiplicar el número de escalares, que como ya se sabe es igual a la dimensión del vector, por la cantidad de bits clásicos asignados para la representación del signo. De manera entonces que la formula se vería de la siguiente manera, donde se va a representar a esta cantidad de bits clásicos como b.

$$b = 1 * 2^n$$

Ecuación 71. Cantidad de bits clásicos para representar los signos de los escalares para una n cantidad de bits cuánticos.

Finalmente, para obtener la cantidad de bits clásicos necesarios para representar los estados puede ponerse un tanto complicado. Recordando que los estados son vectores, y que se asigna un bit a cada uno de los dígitos del vector, por ende, la cantidad de bits necesarios para representar uno de los estados del sistema es igual a la dimensión del estado. Ahora la cantidad de estados es igual también a la dimensión del estado. Ahora se multiplica ambos escenarios de manera de poder obtener la cantidad de bits clásicos necesarios para la representación de los estados. Pudiendo entonces definir la ecuación para la cantidad de bits clásico para los estados de la siguiente manera, donde se representará esta cantidad como c.

$$c = 2^n * 2^n$$

Ecuación 72. Cantidad de bits clásicos necesarios para representar los estados para una n cantidad de bits clásicos.

Entonces, para poder calcular el total de bits necesario para representar todo el sistema seria la suma de todas estas cantidades. Luego, al jugar un poco con la aritmética se puede obtener la siguiente ecuación.

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

Ecuación 73. Cantidad de bits clásicos necesarios para representar una n cantidad de bits cuánticos.

En esta ecuación, n es, como ya se había mencionado, la cantidad de bits cuánticos, y m es la cantidad de bits clásicos. Está claro que esta fórmula es dependiente de la cantidad de bits que se decida usar para representar el bit cuántico. Cuanto mayor sea el valor más preciso son las aproximaciones de los escalares α, β, etc. Se elige representarlo como dieciocho bits ya que no se quiere agotar los recursos de la FPGA con pocos bits cuánticos. En la siguiente ilustración se verá el crecimiento que tiene el número de bits clásicos debido al número de bits cuánticos.

$m = 2^n(7 + 2^n)$	
bits cuánticos (n)	bits clásicos (m)
1	18
2	44
3	120
5	1,248
8	67,328
10	1,055,744
100	1.606938E+60

Tabla 10. Relación entre el bit cuántico y el bit clásico.

Fuente de la imagen: Elaboración propia.

Sin embargo, se quiere controlar las entradas de las representaciones de los bits cuánticos con un solo pulso. Viéndonos en la necesidad de controlar los dos escenarios con un solo bit. El primer escenario es cuando el valor de α es igual a 1 y el de β es igual a 0 y el segundo escenario es el caso contrario. Definiendo ambos escenarios como el set de bits: 0b010000010000000001, para el primer escenario, y 0b000000010010000001, para el segundo escenario. Para esto se usará un multiplexor, que permite controlar el paso de información entre dos escenarios por medio de un pulso.

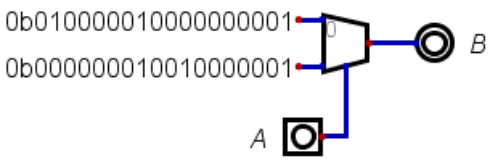


Ilustración 34. Representación de control de un bit cuántico en el software Digital.

Fuente de la imagen: Elaboración propia.

5.1.2 Dos Bits Cuánticos

La tabla 10 de la relación entre los bits cuánticos y clásicos, en base a la representación de un bit cuántico, no crece de manera lineal sino exponencial. Ósea que si se tienen 2 bits cuánticos no es tan fácil como sumar la cantidad de los dos sets de bits clásicos. Ya que el producto tensor entre los estados produce que la cantidad de estados y escalares se duplique. Entonces se tiene que manejar los bits de manera de poder generar este efecto también. Para esto se tiene que crear un circuito que logre unir estos dos bits, ósea que la entrada sea de dos sets de 18 bits y la salida sea de 44 bits. Para eso se tendrá que generar un circuito que logre hacer un producto sensor entre los estados, considerando las cuatro posibilidades para los cuatro estados resultantes, y las multiplicaciones entre los escalares, que de aquí nace el problema más complejo al momento de tratar estos sets de bits debido al lineamiento establecido.

Primero se tendrá que tratar los escalares, para esto se usaran conceptos definidos de manera de poder ahorrar recursos. Esto se refiera a que ya se sabe cuáles deberían de ser los estados de respuesta, debido al producto tensor de los estados de entrada. Por lo que entonces se le asignaran constantes a los valores de salida de los escalares. Sabiendo que estos se representarían con los sets de bits: 0b1000, 0b0100, 0b0010 y 0b0001. Cada uno siendo respectivamente la representación de los estados $|00\rangle$, $|01\rangle$, $|10\rangle$ y $|11\rangle$. Teniendo entonces ya 16 de los 44 bits necesarios para la representación de dos bits cuánticos en el sistema. También se tiene una de las dos funciones necesarias para el circuito que une dos bits cuánticos.

Ahora se tiene que trabajar con la unión de los escalares de cada uno de los bits cuánticos. Si se recapitula la ecuación 33. El escalar que acompaña el estado $|00\rangle$ del sistema de dos bits es la multiplicación del escalar que acompaña el estado $|0\rangle$ del primer bit cuántico por el escalar que acompaña el estado $|0\rangle$ del segundo bit cuántico. De esta manera

se puede definir que el escalar que acompaña un estado de un sistema de dos bits cuánticos es igual a la multiplicación de los escalares de los dos bits cuánticos individuales que acompañan los estados que conforman este nuevo estado de dos bits cuánticos. De manera entonces que se tiene que poder multiplicar los sets de bits entre ellos. Pero los sets de bits no se rigen por la forma tradicional de tratar los bits. Para eso se puede notar que al manejar los bits como un set de bits con asignación convencional al ser divididos esos valores por 32 devuelve el valor que se quiere representar, pero el problema aquí cae en que la FPGA no tiene divisores. Y para este multiplicador especial se tendrá que generar un circuito de cero.

Para poder realizar esto se tiene que resolver un sistema de 12 variables, ya que se debe conseguir multiplicar los dos escalares, ambos siendo un set de 6 bits. Sin embargo, se aprovecharán varias propiedades para poder reducir la cantidad de variables a resolver en el sistema. Se empieza con el concepto de que las magnitudes de los escalares no subirán de 1 y cuando una de ellas tenga este valor la otra, del mismo bit cuántico, será 0. Esta propiedad permite dejar a un lado el bit más significativo. Esto porque este bit más significativo se le asigna el valor de 2^0 esto es igual a 1, y entonces al saber que el bit más significativo es 1 el valor a devolver es igual al otro valor No interesa que valor pudiese llegar a tener el resto de bits, aunque en teoría todos estos bits deberían ser cero ya que el valor nunca sería mayor a 1, pero en qué caso que lo fuese el error se corrige aquí ya que el valor que devuelve es el del otro escalar. En caso que ambos bits más significativos de los dos escalares siendo multiplicados sean 1 el valor a devolver es el set de bits 0b100000.

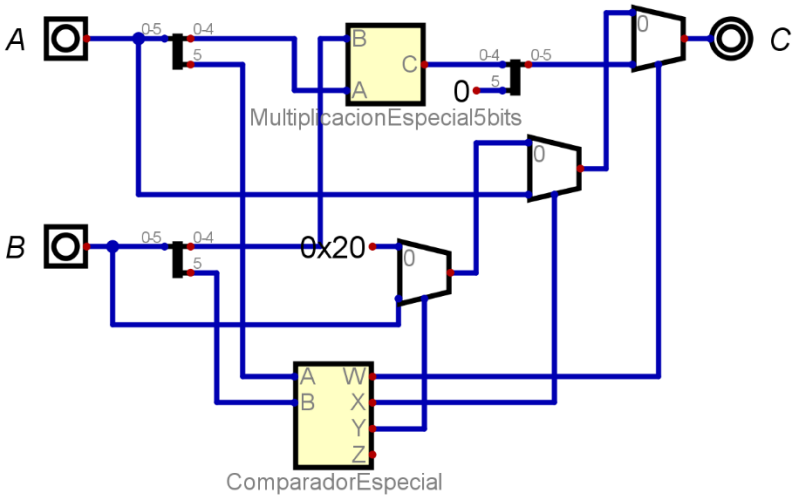


Ilustración 35. Representación del multiplicador especial en el software Digital.

Fuente de la imagen: Elaboración propia.

El circuito Comparador Especial es nada más un circuito que devuelve un uno solamente en la salida W si ambas entradas son 0, devuelve un uno solamente en la salida X cuando solamente la entrada B es uno, devuelve un uno solamente en la salida Y cuando la entrada A es uno y devuelve un uno en la salida Z cuando ambas entradas son uno. Esto ayuda a llevar el control de los bits más significativos, intentando controlar la lógica ya especificada para el manejo de la respuesta dependiendo del valor de este bit.

Ahora, en caso que el bit más significativo sea cero, es cuando llega lo complejo, y es cuando entra el circuito visto en la Ilustración 35, MultiplicaciónEspecial5bits. Para esto se tiene que analizar la respuesta que se obtendría al multiplicar de cada uno de los posibles casos de valores para los escalares, definidos en la tabla 9, entre ellos, ósea multiplicar cada uno por cada uno incluyendo el valor como tal por el mismo. Luego compara este valor debido a la multiplicación y compararlo devuelta con la misma tabla 9 para ver cuál debería de ser el set de bits de respuesta que debe devolver la multiplicación entre los escalares. Para esto se hace una tabla de verdad de 10 variables y se definen 5 bits de respuesta donde estos 5 bits son las respuestas definidas en el análisis ya descrito. Sin embargo, se tiene una gran problemática, y es que la herramienta que se tiene para poder resolver los mapas de Karnaugh tiene una limitante de 8 variables. Para poder adaptar este sistema se toman los bits más significativos de ambos sistemas de 5 variables y se hacen a un lado de manera que se secciona las respuestas en 4 posibles escenarios cuando estos dos bits más significativos son 00, 01, 10 o 11.

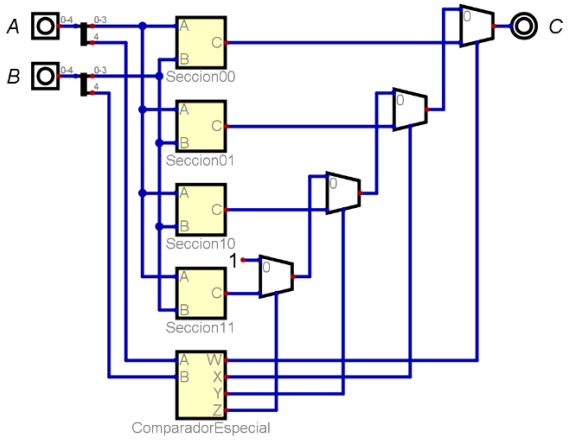


Ilustración 36. Representación del circuito MultipliaciónEspecial5bits en el software Digital.

Fuente de la imagen: Elaboración propia.

Dándonos esto, entonces, la necesidad de tener que generar un circuito lógico digital para cada uno de los 5 bits de respuesta para los cuatro posibles casos de las variables más significativas. Por lo que cada escenario se divide en secciones en base a los parámetros dictados por los bits más significativos de este set de 5 bits. De esta manera entonces se obtuvo estos circuitos que nombras en base a la sección que representan.

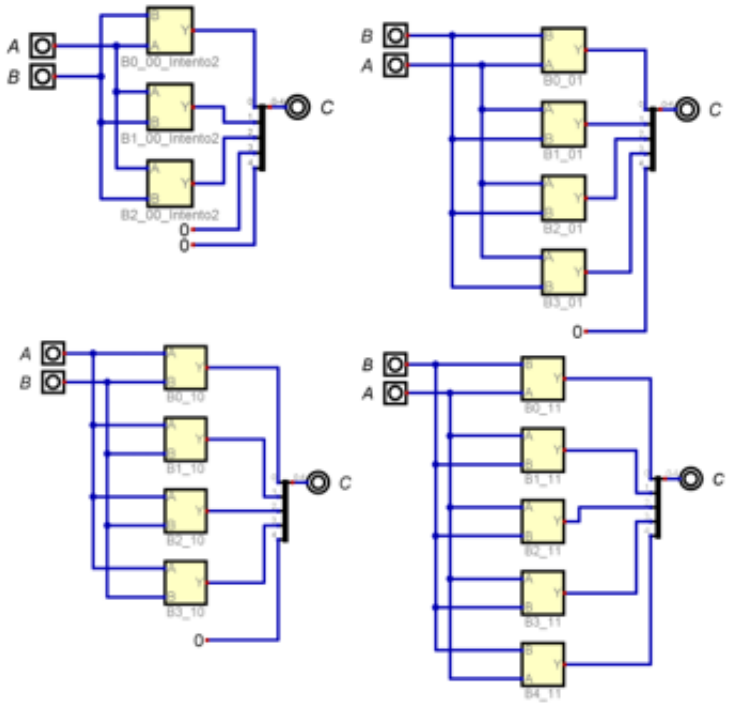


Ilustración 37. Representaciones de las cuatro secciones en el software Digital.

Fuente de la imagen: Elaboración propia.

Estos circuitos están conformados de cada circuito necesario para cada bit de la respuesta como ya se había descrito. Estos circuitos se realizaron por medio del software Karnaugh Map Minimizer. Dando como resultado ecuación booleanas de hasta 50 expresiones de ANDs unidas por un OR, esto debido a que la resolución se hizo por medio de miniterminos.

Finalmente, para poder definir el circuito digital para la fusión de dos bits cuánticos se tiene que definir también los signos de estos. Sabiendo entonces que en cada caso que se multiplique se tendrán cuatro posibles escenarios de signos. Sabiendo entonces la ley de los signos se sabe que más por más es igual a más, menos por más es igual a menos, y menos por menos es igual a más. Si el bit del signo es 0 el escalar es positivo y si es 1 es negativo.

Esto significa que, al multiplicar dos escalares, el juego con los signos se puede analizar de tal forma que si ambos casos son iguales se quiere que la respuesta a devolver sea 0, ya que, si ambos son iguales, ambos son negativos o positivos, por lo que como ya se vio la multiplicación de estos casos es positiva que se tiene definido como 0. Esto es igual al circuito lógico XOR que devuelve un uno solo si ambas entradas son diferentes. Por lo que se aplicará un XOR para cada uno de los signos de los escalares siendo multiplicados.

Teniendo entonces ya definido el circuito que se estará utilizando para la multiplicación de los bits de los escalares, bajo los parámetros para la representación de los bits cuánticos, y el circuito necesario para la comparación de los signos se puede terminar el circuito para la fusión de dos bits cuánticos. Este lo se aplica en la multiplicación que se había explicado para los escalares. Para dejar claro este concepto verse la tabla 3. De manera que el circuito necesario para la fusión de dos bits cuánticos es el siguiente.

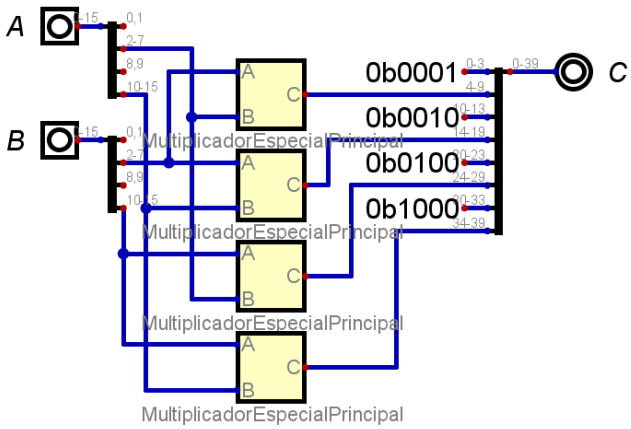


Ilustración 38. Representación del circuito para la fusión de dos bits cuánticos.

Fuente de la imagen: Elaboración propia.

5.1.3 Tres Bits Cuánticos

El tercer ciclo de la metodología será lograr aplicar el algoritmo de Deutsch para el problema Deutsch Jozsa. La diferencia entre este y el problema original es que es este problema es un mapeo de una n cantidad de bits cuánticos, n siendo mayor a 1, a un bit cuántico. Ahora el crecimiento será aplicar tres bits cuánticos. Por lo que aún no se tienen las herramientas necesarias para la función de estos. Se estará aplicando dos formas de esta herramienta. Una donde la entrada sea para tres bits cuánticos por separados, ósea que hablando en términos de la representación se va a tener tres entradas de dieciséis bits

clásicos. La segunda se tendrán 2 entradas. Una seria donde en la entrada el bit más significativo está solo, ósea que esta entrada es para dieciséis bits. La otra sería una entrada donde los siguientes dos bits ya fueron fusionados, ya sea por la herramienta descrita en la sección anterior, una compuerta CNOT o ContraCNOT, ósea que esta entrada será de cuarenta y cuatro bits.

Primero se trabajará con la fusión de los tres bits cuánticos por separados, que se llamarán ThreeQubits por su nombre en inglés. En esta se tendrán las tres entradas de dieciséis bits se separan los bits en estados escalares y les darán sus propias entradas a los signos, separando a estos en algún proceso previo. Entonces primero se trabajará con los escalares. Multiplicando cada uno por el necesario para los escalares de respuesta. Se debe recordar que estos son un producto escalar entre ellos, (Marinescu & Marinescu, 2012).

$$\begin{pmatrix}\alpha_A\\ \beta_A\end{pmatrix}\otimes\begin{pmatrix}\alpha_B\\ \beta_B\end{pmatrix}\otimes\begin{pmatrix}\alpha_C\\ \beta_C\end{pmatrix}=\begin{pmatrix}\alpha_A\alpha_B\\ \alpha_A\beta_B\\ \beta_A\alpha_B\\ \beta_A\beta_B\end{pmatrix}\otimes\begin{pmatrix}\alpha_C\\ \beta_C\end{pmatrix}=\begin{pmatrix}\alpha_A\alpha_B\alpha_C\\ \alpha_A\alpha_B\beta_C\\ \alpha_A\beta_B\alpha_C\\ \alpha_A\beta_B\beta_C\\ \beta_A\alpha_B\alpha_C\\ \beta_A\alpha_B\beta_C\\ \beta_A\beta_B\alpha_C\\ \beta_A\beta_B\beta_C\end{pmatrix}$$

De manera entonces que se puede representar cada escalar de respuesta como una multiplicación entre los escalares de entrada de la siguiente manera.

$$\alpha=\alpha_A\alpha_B\alpha_C,\qquad \beta=\alpha_A\alpha_B\beta_C,\qquad \gamma=\alpha_A\beta_B\alpha_C,\qquad \delta=\alpha_A\beta_B\beta_C$$

$$\varepsilon=\beta_A\alpha_B\alpha_C,\qquad \zeta=\beta_A\alpha_B\beta_C,\qquad \eta=\beta_A\beta_B\alpha_C,\qquad \theta=\beta_A\beta_B\beta_C$$

Utilizando estas expresiones y aplicando el multiplicador especial, explicado en la sección anterior, se puede entonces definir cada uno de los escalares de salida. Sin embargo, el multiplicador especial es de dos entradas por lo que se tiene que adaptar. Para esto se utilizará dos multiplicadores para cada uno de los productos. Tomando de ejemplo entonces el primer producto descrito, se aplicará esta multiplicación de la siguiente manera.

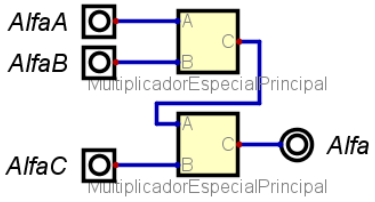


Ilustración 39. Producto de tres escalares en Digital.

Se aplicará esto a cada uno de los productos de manera de obtener los ocho escalares de respuesta. Luego estos serán unidos a sus respectivos estados, |000⟩, |001⟩, |010⟩, |011⟩, |100⟩, |101⟩, |110⟩ y |111⟩, respectivamente, utilizando un bit clásico para cada uno de los dígitos del vector que conforma estos estados.

Ahora se trabajará los signos de estos productos, utilizando una entrada individual para cada uno de los seis bits de signos entre los tres bits cuánticos. Para esto se aplicará la misma lógica para todos los casos y luego se distribuye la información entre los escalares descritos en los productos vistos en las expresiones anteriores. Ahora se sabe que los signos, en la representación asignada a los bits cuánticos, de los escalares son positivos si este bit es 0 y negativos si este bit es 1. Entonces sabiendo entonces las leyes de signos, donde la respuesta es positiva si el número de negativos es par o cero. Esto visto desde un punto de vista lógico, aplicando los tres signos que se estará multiplicando simultáneamente, tiene que ser un circuito que devuelva 1 si el número de unos en la entrada es impar. De manera entonces que se puede aplicar la siguiente tabla de verdad.

A	B	C	Y
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

Tabla 11. Tabla de verdad para un producto de signos entre tres variables.

Fuente de la imagen: Elaboración propia.

Aplicando entonces el procedimiento del mapa de Karnaugh para la resolución de esta tabla de verdad, (Karnaugh, 1953). Teniendo entonces como respuesta, en base a la ecuación booleana, se obtiene el siguiente circuito.

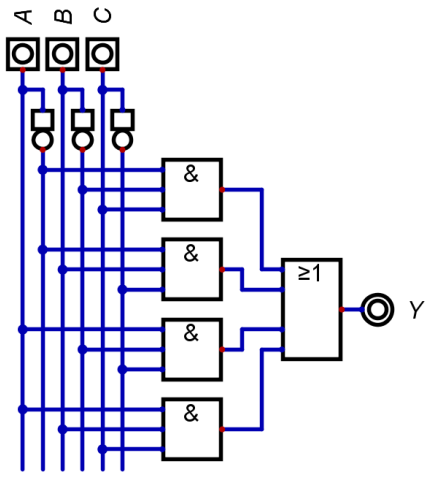


Ilustración 40. Circuito multiplicador de signos de tres valores.

Fuente de la imagen: Elaboración propia.

Sabiendo entonces como multiplicar tanto los escalares como sus signos se puede proceder a diseñar el circuito necesario para la fusión de tres bits cuánticos por separado. De tal manera que uniendo todos estos conceptos se obtiene el siguiente circuito.

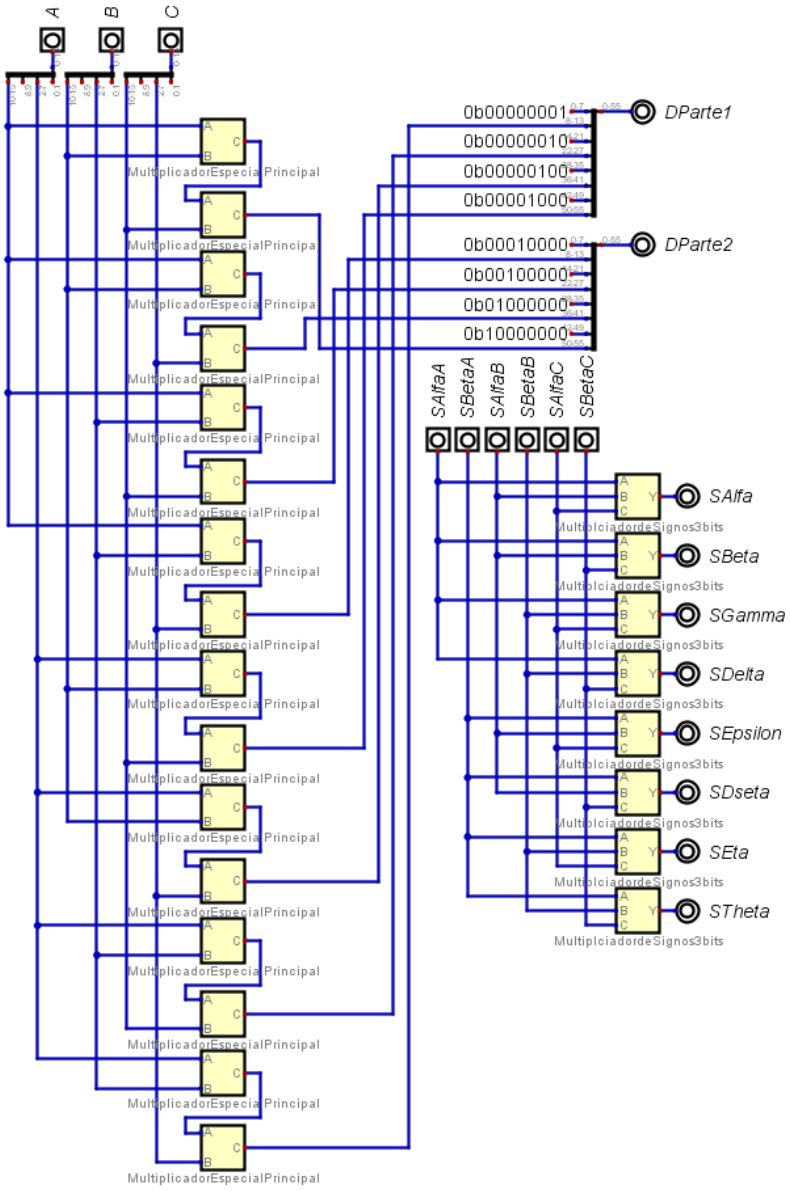


Ilustración 41. Circuito para la fusión de tres bits cuánticos en Digital.

Fuente de la imagen: Elaboración propia.

Como se puede ver hay un detalle importante en el diseño del circuito y es que los bits aceptables por bus son de 64. Entonces, recordando la tabla 10, 3 bits cuánticos son un conjunto de 120 bits clásicos de manera entonces que se va a dividir la salida en dos partes cada una de 56 bits de manera también que se manejarán los bits de los signos con sus propias salidas.

Ahora se debe considerar el segundo caso necesario para el manejo de tres bits cuánticos, que es el caso en el que se tienen dos entradas, una de dieciséis bits para un bit cuántico y una de cuarenta y cuatro bits para dos bits cuánticos ya unidos. En este caso

también se empieza manejando los escalares. Para lo que entonces se aplicará el producto tensor para un sistema de un bit cuántico por un sistema de dos bits cuántico. De manera de obtener los productos necesarios de manejar para los escalares de respuesta.

$$\begin{pmatrix}\alpha_A\\ \beta_A\end{pmatrix}\otimes\begin{pmatrix}\alpha_B\\ \beta_B\\ \lambda_B\\ \delta_B\end{pmatrix}=\begin{pmatrix}\alpha_A\alpha_B\\ \alpha_A\beta_B\\ \alpha_A\lambda_B\\ \alpha_A\delta_B\\ \beta_A\alpha_B\\ \beta_A\beta_B\\ \beta_A\lambda_B\\ \beta_A\delta_B\end{pmatrix}$$

De manera entonces que se puede representar cada escalar de respuesta como una multiplicación entre los escalares de entrada de la siguiente manera.

$$\alpha=\alpha_A\alpha_B,\qquad \beta=\alpha_A\beta_B,\qquad \gamma=\alpha_A\lambda_B,\qquad \delta=\alpha_A\delta_B$$

$$\varepsilon=\beta_A\alpha_B,\qquad \zeta=\beta_A\beta_B,\qquad \eta=\beta_A\lambda_B,\qquad \theta=\beta_A\delta_B$$

Se aplicará esto a cada uno de los productos de manera de obtener los ocho escalares de respuesta. Luego estos serán unidos a sus respectivos estados, $|000\rangle$, $|001\rangle$, $|010\rangle$, $|011\rangle$, $|100\rangle$, $|101\rangle$, $|110\rangle$ y $|111\rangle$, respectivamente, utilizando un bit clásico para cada uno de los dígitos del vector que conforma estos estados.

Ahora se estará trabajando con los signos, a los cuales se le asignará una entrada a cada uno. Ahora en este caso la multiplicación de los signos será más fácil, ya que solo se estará trabajando con un producto entre dos escalares. Dicho esto, entonces se puede aplicar la misma lógica que se utiliza para la multiplicación de signos en el circuito para la unión de dos bits cuánticos. Esto significa que dependiendo del escalar del cual se estará consiguiendo el signo se usará un XNOR para los dos signos de entrada de los escalares que lo conforman. El circuito para fusión de 3 bits cuánticos, con dos entradas una para un bit cuántico y otra para dos bits cuánticos, es el siguiente.

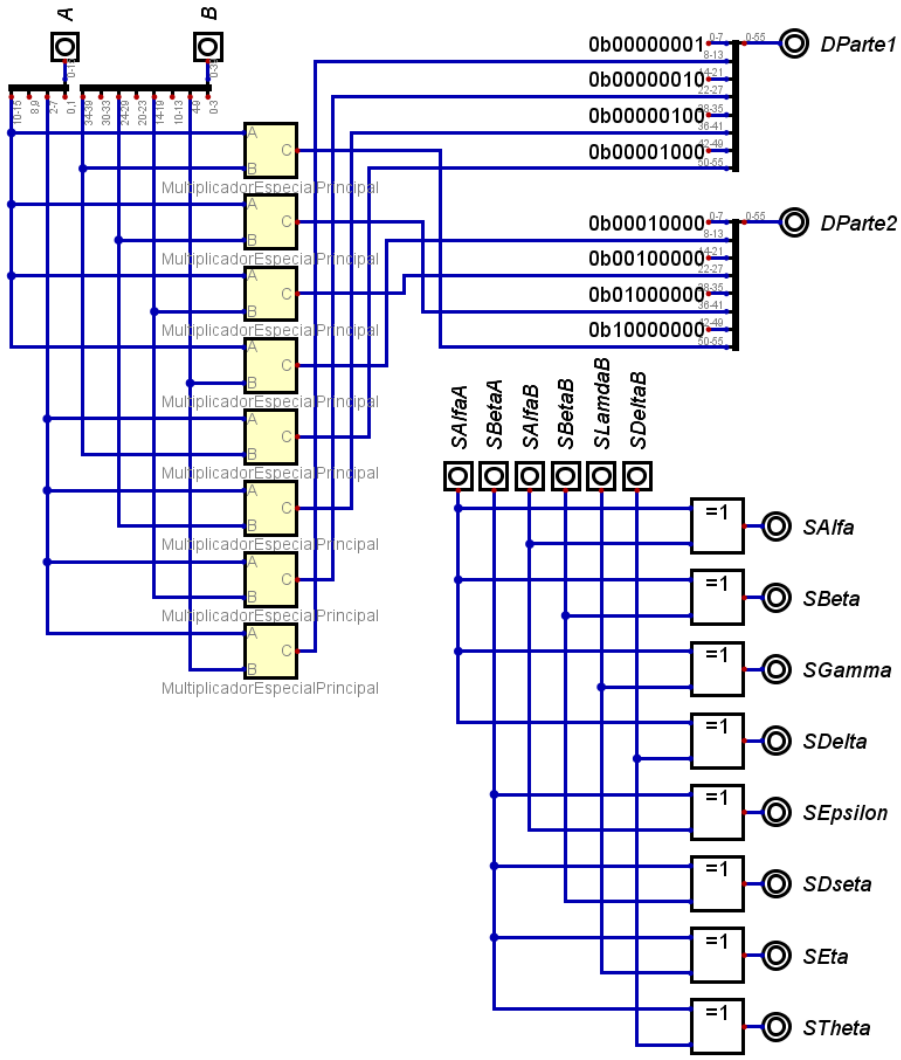


Ilustración 42. Circuito para la unión de tres bits cuánticos de dos entradas en Digital.

Fuente de la imagen: Elaboración propia.

5.2 Primer Ciclo

Se estará aplicando el algoritmo de Deutsch. Para este algoritmo, si se recapitula la sección 3.1.12, se estarán utilizando las compuertas NOT, CNOT y ContraCNOT.

5.2.1 Análisis de las Compuertas Cuánticas

En esta etapa del primer ciclo se estará analizando las compuertas cuánticas de manera de poder sintetizar sus representaciones matemáticas y saber las herramientas necesarias para poder representar las compuertas cuánticas.

Primero se estará analizando la compuerta cuántica de Haddamard, probablemente la compuerta más esencial para este algoritmo. Si se recuerda de la sección 3.1.10, la

compuerta de Haddamard busca rotar un bit cuántico. Asi pudiendo agarrar un bit cuántico con un estado definido y poder hacer que este esté en un estado de superposición. Matemáticamente hablando la compuerta lo que hace es que haya un juego aritmético entre los escalares, tal operación afectando ambos estados. Ahora la compuerta afecta un vector de la siguiente manera, donde se va a utilizar un bit cuántico en su función convencional de estado de superposición, rescribiendo esta fórmula de superposición como un solo vector.

$$H|x\rangle = \frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix}\begin{pmatrix}\alpha\\ \beta\end{pmatrix} = \begin{pmatrix}\frac{\alpha+\beta}{\sqrt{2}}\\ \frac{\alpha-\beta}{\sqrt{2}}\end{pmatrix} = \frac{\alpha+\beta}{\sqrt{2}}|0\rangle + \frac{\alpha-\beta}{\sqrt{2}}|1\rangle$$

Entonces si se puede observar, matemáticamente la compuerta lo que hace es sumar o restar y luego dividir entre la raíz cuadrada de 2. En esta operación se tienen tres problemas esenciales para poder resolver esta compuerta cuántica en términos clásicos. El primero es que el FPGA que se estará utilizando, el SPARTAN 6, no tiene circuitos divisores. El segundo, es el juego entre los signos. Sin embargo, si se pone a sumar estos números como tales ellos van a asumir ese primer bit como parte del entero. Que aquí es donde cae el tercer problema, el circuito sumador o restador va a asumir el numero como un entero y la representación es tomar estos valores como decimales, ya que el número de los escalares no será mayor a uno.

Ahora el primer problema que se debe de resolver es el de poder dividir entre la raíz cuadrada de dos. Para esto se tendrá que realizar un circuito capaz de poder procesar la información de entrada y devolvernos el set de bits que, en teoría, es el set de bits que debería de devolver debido a la división de un número, bajo las lineaciones, entre la raíz cuadrada de dos. Para esto se realiza una tabla de verdad de 6 variables donde la respuesta era el valor que devolvía la división del valor de entrada entre la raíz cuadrada de dos. Se parametriza cada valor para que devolviera el set de bits en base a la tabla 9.

Una vez planteado esto se tiene que ser capaces de poder restar o sumar debido al respectivo set de signos que van a tener los dos escalares. Para esto se tendrá que jugar con la aritmética entre los valores y considerar las cuatro posibilidades de signos que se tienen, +α, +β; +α, -β; -α, +β; y -α, -β. Para esto se considerará los cuatro casos que se analizarán uno por uno.

Caso 1:

En el caso que ambos sean positivos, se trata la aritmética tal y como se ve en el análisis que se le hizo a la formula ya planteada. Les se estará aplicando la división por la raíz cuadrada previo a la aritmética a los escalares, esto debido a que en los siguientes casos se estará tomando estos valores ya divididos desde este punto de manera de ahorrar recursos. Para el primer escalar de la respuesta se tiene que sumar los valores nada más. Ahora en el caso de la resta se tendrán dos situaciones a considerar, y son si $\alpha > \beta$ o $\alpha < \beta$. La importancia detrás de esto recae en los signos, ya que en caso que $\alpha < \beta$ la resta se vuelve un numero negativo. Pero en el otro caso no importa ya que sigue siendo positivo. Lo que se hará entonces en base al segundo escenario, el cual en vez de hacer la resta α menos β, se hará la resta β menos α y se dice que el signo del nuevo β sea negativo, ósea que se enciende el bit.

Caso 2:

En el caso que solamente el β de entrada sea negativo se tiene que hacer un ajuste en cuanto ambos escalares de respuesta. En el caso del nuevo α este es la suma del α y β de entrada. Pero como β es negativo se tiene que tratar este escalar ya no como una suma sino como una resta entre los escalares de entrada, en el cual también se tiene que considerar la comparación si $\alpha > \beta$ o $\alpha < \beta$. En este caso, si $\alpha < \beta$ se estará devolviendo que el signo del nuevo α es negativo por lo que se encenderá el bit para el signo de este escalar. En la parte del nuevo β se tiene como una resta de $\alpha - \beta$, y como β es negativo al multiplicar signos queda una simple suma. Por lo que el nuevo β es igual a la suma de los escalares de entrada, claramente todo esto ya ha sido procesado por el circuito divisor entre la raíz cuadrada de dos.

Caso 3:

En el caso que solamente el α de entrada sea negativo se tiene que hacer unos ajustes en la aritmética para los escalares de respuesta. Para el nuevo α si se recuerda es una suma entre α y β de entrada, por lo que en caso que α sea negativo se tiene que plantear los dos mismos casos que se han estado aplicando para las restas cuando $\alpha > \beta$ o cuando $\alpha < \beta$. Con la diferencia que el signo del nuevo α va a depender de que si $\alpha > \beta$ ya que cuando esta condición se cumpla este signo será negativo. En el caso del β de salida depende de la

resta $\alpha - \beta$ por lo que en si α es negativo se puede sacar como factor común el signo negativo y dejar expresado esto como una suma. En este caso no importa cuál sea mayor que el otro ya que en ambos casos el nuevo β tiene que ser negativo.

Caso 4:

En el último caso se tendrá que ambos escalares de entrada son negativos, por lo que se tendrá que siempre hacer ajustes, pero leves. En este caso la primera operación de suma, como se está sumando dos numero negativos se puede sacar de factor común el negativo y nada más realizar la suma entre los dos escalares, y asumir por default que en este caso el signo del nuevo α es negativo. Ahora para el nuevo β se tiene que realizar una resta para la cuál como el β de entrada es negativo al multiplicarlo por el menos se vuelve negativo y se tiene la nueva resta que seria $\beta - \alpha$. Donde también se tiene que hacer la comparación si $\alpha > \beta$ o $\alpha < \beta$, en caso que $\alpha > \beta$ se asume que el signo para el nuevo β es negativo.

La siguiente compuerta que se analizará será la compuerta NOT. Esta compuerta conserva la misma lógica que su contraparte clásica, cambiando de estado un bit cuántico. Ese es el concepto que se maneja, pero sin embargo importa el impacto matemático que esta tiene sobre los bits cuánticos. Se necesita entonces sintetizar la fórmula para poder hacer el análisis de la reacción o cambios que pueda tener la expresión del bit cuántico.

$$X|x\rangle = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} \beta \\ \alpha \end{pmatrix} = \beta|0\rangle + \alpha|1\rangle$$

En base a esta expresión se puede entonces definir que lo que la compuerta NOT hace sobre un bit cuántico es intercambiar los valores de los escalares de cada estado, claramente esto incluye el signo del bit cuántico.

La siguiente compuerta que se analizará será la compuerta CNOT. Esta compuerta funciona sobre dos bits cuánticos, donde uno es de control y el otro es el controlado. Cuando el bit de control está en estado $|1\rangle$ el bit controlado cambia de estado. Sin embargo, como en las otras compuertas, interesa ver el impacto matemático que este tiene sobre la expresión del bit cuántico en superposición, o en este caso de los dos bits cuánticos. Se define el sistema de dos bits cuánticos con la siguiente expresión.

$$|xy\rangle = \alpha|00\rangle + \beta|01\rangle + \lambda|10\rangle + \delta|11\rangle = \begin{pmatrix} \alpha \\ \beta \\ \lambda \\ \delta \end{pmatrix}$$

La matriz que representa la compuerta CNOT de la ecuación 45, se aplicará sobre la expresión definida para el sistema de dos bits cuánticos. Se hará esto de manera de poder sintetizar la formula en una manera que se pueda expresar en los circuitos.

$$CNOT|xy\rangle = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \\ \lambda \\ \delta \end{pmatrix} = \begin{pmatrix} \alpha \\ \beta \\ \delta \\ \lambda \end{pmatrix} = \alpha|00\rangle + \beta|01\rangle + \delta|10\rangle + \lambda|11\rangle$$

Viendo esto se puede entonces deducir que la compuerta CNOT lo que matemáticamente hace es intercambiar el escalar entre los estados $|10\rangle$ y $|11\rangle$. Sin embargo, también es de importancia saber el escenario en el que el primer bit cuántico es el bit de control. En este caso la matriz que representa la compuerta cuántica tiene un pequeño cambio que, ya afectando el vector planteado para el sistema de dos bits cuánticos, se ve de la siguiente manera.

$$CNOT|xy\rangle = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \\ \lambda \\ \delta \end{pmatrix} = \begin{pmatrix} \alpha \\ \delta \\ \lambda \\ \beta \end{pmatrix} = \alpha|00\rangle + \delta|01\rangle + \lambda|10\rangle + \beta|11\rangle$$

Dándonos esta expresión el poder de deducir que en este caso quienes cambian de escalar son los estados $|01\rangle$ y $|11\rangle$. Es importante poder saber cómo controlar el bit control, valga la redundancia, ya que se le quiere dar una vista y un diseño lógico para los circuitos clásicos general de los posibles casos que se e pueden dar a esta compuerta cuántica.

Ahora se verá la compuerta CNOT desde otra perspectiva, cuando este está trabajando con tres bits cuánticos simultáneamente. En este caso se aplicará de dos formas, cuando el bit más significativo es el de control y los otros dos son los controlados y cuando el de control es el segundo más significativo y los otros dos son los controlados. De tal forma entonces que aplicando la representación matemática de la matriz de la compuerta CNOT en un sistema de tres bits cuánticos, donde el bit significativo es el de control y los otros dos bits cuánticos son los controlados, se obtiene el siguiente resultado.

$$CNOT|xyz\rangle = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \\ \gamma \\ \delta \\ \varepsilon \\ \zeta \\ \eta \\ \theta \end{pmatrix} = \begin{pmatrix} \alpha \\ \beta \\ \gamma \\ \delta \\ \theta \\ \eta \\ \zeta \\ \varepsilon \end{pmatrix}$$

Viendo esto entonces se sabe que los escalares cambian entre los estados $|100\rangle$ y $|111\rangle$, y entre los estados $|101\rangle$ y $|110\rangle$. También buscando que los signos de los escalares se muevan con ellos, siempre teniendo sus propias entradas y salidas. Ahora aplicando la matriz de la compuerta CNOT, pero bajo la condición que el bit cuántico de control es el segundo más significativo y los otros dos son los controlados se obtiene la siguiente igualdad.

$$CNOT|xyz\rangle = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \\ \gamma \\ \delta \\ \varepsilon \\ \zeta \\ \eta \\ \theta \end{pmatrix} = \begin{pmatrix} \alpha \\ \beta \\ \theta \\ \eta \\ \varepsilon \\ \zeta \\ \delta \\ \gamma \end{pmatrix}$$

Viendo esto entonces se sabe que los escalares cambian entre los estados $|010\rangle$ y $|111\rangle$, y entre los estados $|011\rangle$ y $|110\rangle$. Cambiando los signos también de los escalares a estas mismas posiciones.

La ultima compuerta que se analizará es la compuerta ContraCNOT. Esta compuerta tiene la misma funcionalidad que la compuerta CNOT con la pequeña diferencia que el cambio en el bit controlado se produce cuando el bit de control es 0. De manera que al sintetizar se observa la siguiente expresión.

$$ContraCNOT|xy\rangle = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \\ \lambda \\ \delta \end{pmatrix} = \begin{pmatrix} \beta \\ \alpha \\ \lambda \\ \delta \end{pmatrix} = \beta|00\rangle + \alpha|01\rangle + \lambda|10\rangle + \delta|11\rangle$$

Se ve entonces que los estados que cambian de escalares son $|00\rangle$ y $|01\rangle$ de manera que esto será lo que se quiere ver plasmado en los circuitos digitales. Pero ahora, a la igual que en la compuerta CNOT, se busca también tener control sobre el bit de control, de manera de poder elegir si se quiere que sea el primer o segundo bit. Para poder hacer esto se tiene que entonces también ver la resolución que se le puede dar a la expresión, sintetizando la ecuación en base al primer bit como el de control. De manera que se obtiene la siguiente expresión.

$$ContraCNOT|xy\rangle = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \\ \lambda \\ \delta \end{pmatrix} = \begin{pmatrix} \lambda \\ \beta \\ \alpha \\ \delta \end{pmatrix} = \lambda|00\rangle + \beta|01\rangle + \alpha|10\rangle + \delta|11\rangle$$

Viendo entonces que en este caso los cambios de escalares se dan entre los estados $|00\rangle$ y $|10\rangle$. Queriendo entonces que los bits de los escalares cambien para este set de estados.

Finalmente se estará dándole a la compuerta ContraCNOT la misma aplicación para tres bits cuánticos que a la compuerta CNOT. De manera que también se aplicará el concepto de darle dos casos distintos donde el bit cuántico más significativo es el de control y los otros dos son los controlados, y cuando el segundo bit cuántico más significativo es el de control y los otros dos son los controlados.

De tal forma que entonces aplicando la matriz de ContraCNOT aplicada al primer caso afectando un sistema de tres bits cuánticos se puede obtener la siguiente expresión.

$$ContraCNOT|xyz\rangle = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \\ \gamma \\ \delta \\ \varepsilon \\ \zeta \\ \eta \\ \theta \end{pmatrix} = \begin{pmatrix} \delta \\ \gamma \\ \beta \\ \alpha \\ \varepsilon \\ \zeta \\ \eta \\ \theta \end{pmatrix}$$

De esta forma se sabe entonces que los escalares que cambian son el escalar de $|100\rangle$ con el de $|111\rangle$, y entre el escalar de $|101\rangle$ y el escalar de $|110\rangle$. De tal forma que también se busca que los signos se muevan con sus escalares. Ahora para el segundo caso planteado para esta aplicación de la compuerta cuántica ContraCNOT. De manera que aplicando la matriz en base a la condición de este caso se obtiene la siguiente expresión.

$$ContraCNOT|xyz\rangle = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \\ \gamma \\ \delta \\ \varepsilon \\ \zeta \\ \eta \\ \theta \end{pmatrix} = \begin{pmatrix} \zeta \\ \epsilon \\ \gamma \\ \delta \\ \beta \\ \alpha \\ \eta \\ \theta \end{pmatrix}$$

De esta forma se sabe entonces que los escalares que cambian son el escalar de $|000\rangle$ con el de $|101\rangle$, y entre el escalar de $|001\rangle$ y el escalar de $|100\rangle$. De tal forma que también se quiere que los signos se muevan con sus escalares.

5.2.2 Diseño de los Circuitos Equivalentes

En esta sección se estará planteando los diseños de los circuitos digitales necesarios para poder simular las compuertas. Estos diseños se basarán en los análisis hechos en la sección anterior. Se harán los circuitos de tal manera de poder tener un esquema generalizada para cada una de las compuertas cuánticas ya explicadas y sintetizadas a la necesidad. El planteamiento de los diseños los se harán respectivamente en el orden en el que se realiza el análisis.

Se comenzó con la compuerta de Haddamard, recordando que, para poder entrar al análisis de la aritmética necesaria para simular los posibles casos, con respecto a los signos de entrada, se tuvo que haber dividido los escalares entre la raíz cuadrada de dos. Para eso se tiene que plantear una tabla de verdad con cada caso para los sets de bits de entrada y tener un set de bits, del mismo tamaño, pero mapeados a lo que hubiese sido la división de estos valores de entrada por la raíz cuadrada de dos, está describiéndola en binario bajo los parámetros para la representación de los bits cuánticos. De manera que, por razones de ejemplificar, ciertos casos se mapean de la siguiente manera.

$$000101 \rightarrow 000100, \qquad 010011 \rightarrow 001101$$

Sin embargo, hay un caso que es necesario plantear, y es cuando el bit más significativo de los 6 bits de entrada es 1, debido a que si este bit es 1 la magnitud del escalar es igual 1, en teoría no se debería de tener ningún otro bit encendido. En caso que existiera tal error que algún otro bit entra encendido aparte del más significativo se mapeara el set de bit de la siguiente manera.

$$1xxxxx \rightarrow 010111$$

Se toma el resto de bits como don't care bajo el análisis lógico digital de manera de que sin importar el resto de bits. El set de bits que se devuelve es exactamente el set de bits que representan, ya que es el valor más aproximado, la magnitud $1/\sqrt{2}$. De tal manera entonces que al realizar el mapa de Karnaugh, utilizando el software Karnaugh Map Minimizer, devuelve las siguientes expresiones para cada bit de respuestas. Donde se representará el set de bits de entrada como:

$$A = a_5 a_4 a_3 a_2 a_1 a_0$$

Y los bits de salida los se representarán de la siguiente manera:

$$B = b_5 b_4 b_3 b_2 b_1 b_0$$

De manera entonces que bajo estos parámetros las ecuaciones booleanas para la generación de los circuitos se ven de la siguiente manera.

$$b_5 = 0$$

$$b_4 = a_4 a_3 + a_4 a_2 a_1 a_0$$

$$b_3 = !\,a_4 a_3 a_2 + !\,a_4 a_3 a_1 a_0 + a_4 !\,a_3 a_1 + a_4 !\,a_3 !\,a_2 + a_4 !\,a_3 a_1 !\,a_0$$

$$b_2 = !\,a_4 !\,a_3 a_2 a_0 + !\,a_4 !\,a_3 a_2 a_1 + !\,a_4 a_3 !\,a_2 !\,a_1 + !\,a_4 a_3 !\,a_2 !\,a_0 + a_4 !\,a_3 !\,a_2 a_0 + a_4 !\,a_3 a_1 !\,a_0 + a_4 a_2 !\,a_1 + a_4 a_3 a_2$$

$$b_1 = !\,a_4 !\,a_3 !\,a_2 a_1 a_0 + !\,a_4 !\,a_3 a_2 !\,a_1 !\,a_0 + !\,a_4 a_3 a_2 a_1 + !\,a_4 a_3 !\,a_2 !\,a_1 + a_3 !\,a_2 a_1 !\,a_0 + a_4 !\,a_3 !\,a_1 !\,a_0 + a_4 !\,a_3 a_2 !\,a_1 + a_4 !\,a_3 a_2 a_1 !\,a_0 + a_4 a_3 a_1 a_0 + a_4 a_3 !\,a_2 a_0$$

$$b_0 = !\,a_4 !\,a_3 a_2 !\,a_1 !\,a_0 + !\,a_4 !\,a_3 !\,a_2 !\,a_1 a_0 + !\,a_4 a_3 a_2 a_0 + !\,a_4 a_2 a_1 a_0 + !\,a_4 !\,a_2 a_1 !\,a_0 + a_4 !\,a_2 !\,a_1 a_0 + a_4 a_2 !\,a_1 a_0 + a_4 !\,a_2 a_1 a_0 + a_4 !\,a_3 !\,a_2 a_1 + a_4 a_2 a_1 !\,a_0$$

Los circuitos los se generan en base a los 5 bits menos significativos, de manera de considerar el caso como se planteó cuando el bit más significativo es 1 plantear un solo caso. De manera que si se aplican las ecuaciones booleanas anteriores para los casos en que el bit más significativo es 1. Se controla el valor de salida con un multiplexor de manera que el control lo realiza el bit más significativo. Cuando este es 0 devuelve la respuesta generada por las ecuaciones y cuando es 1 devuelve la constante 0b010111. De tal forma que el circuito se ve de la siguiente manera.

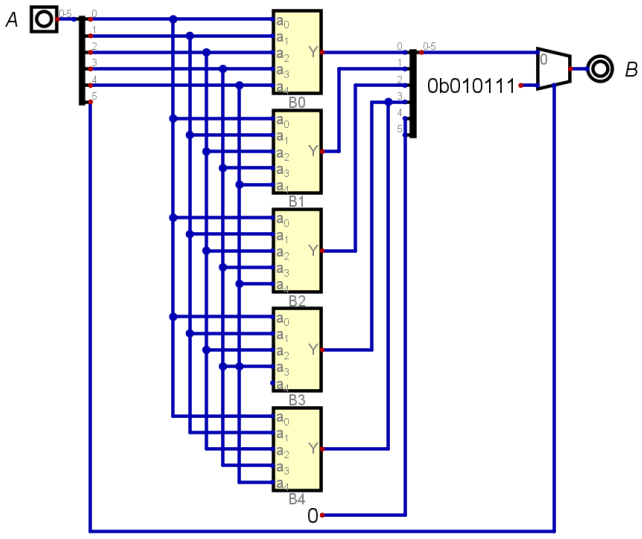


Ilustración 43. Representación del circuito divisor entre la raíz cuadrada de dos en Digital.

Fuente de la imagen: Elaboración propia

Donde A es la entrada de 6 bits del valor del escalar que se busca multiplicar y B es la salida de ese valor ya dividido y mapeado a los parámetros.

Teniendo ya esto entonces se puede proceder a diseñar el circuito para la compuerta de Haddamard. Recordando entonces el juego aritmético planteado en el análisis realizado en la sección anterior, se puede diseñar el circuito de tal forma que se miraría de la siguiente manera.

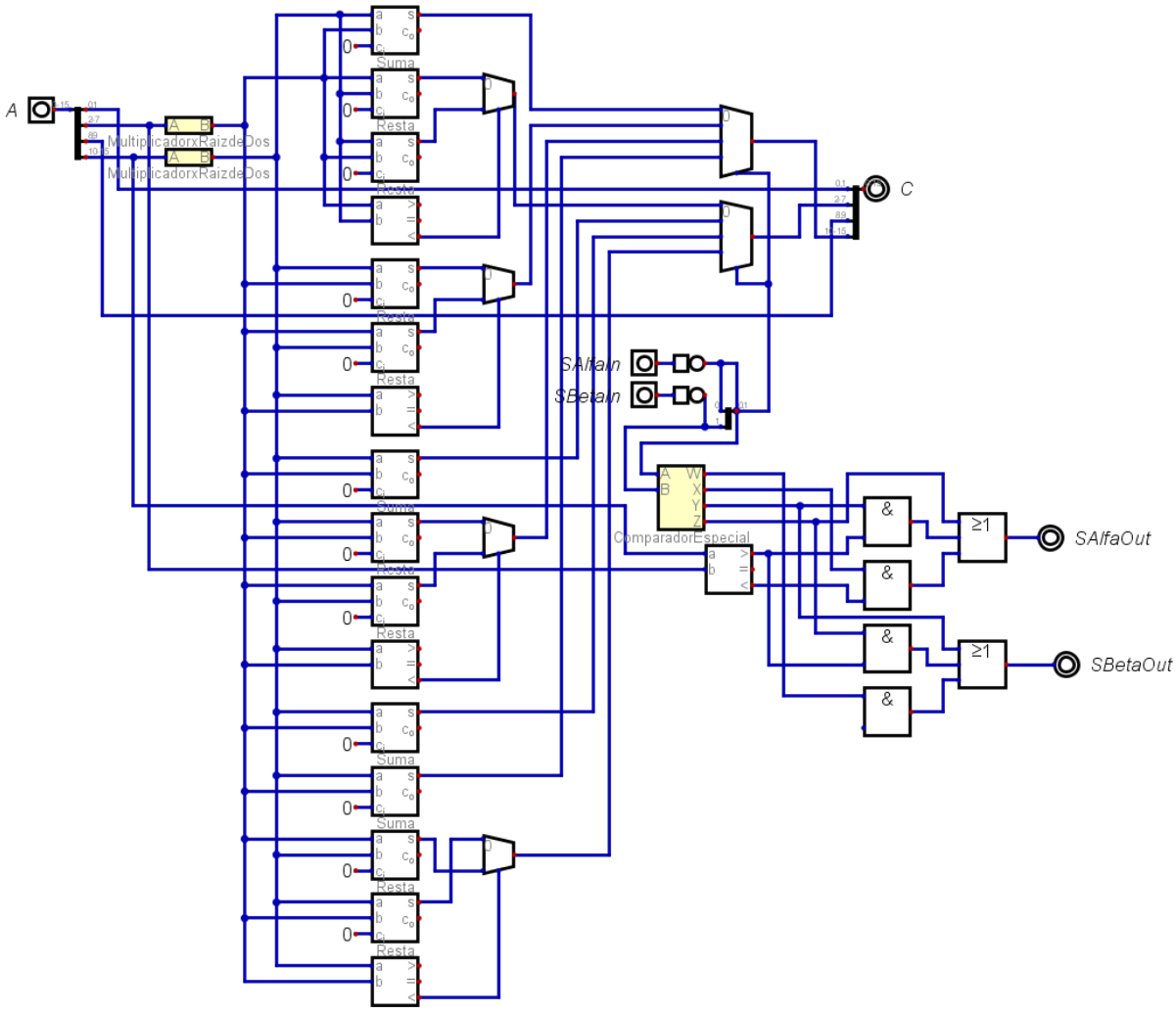


Ilustración 44. Representación de la compuerta Haddamard en Digital.

Fuente de la imagen: Elaboración propia.

El siguiente diseño que se estará planteando es la compuerta NOT que lo que hace esta compuerta es intercambiar el valor de los escalares, con todo y signo. Para esta función no se necesita ningún tipo de aritmética, solo se ocupa cambiar de puestos los cables que portan esta información. De manera que la compuerta NOT se ver de la siguiente forma.

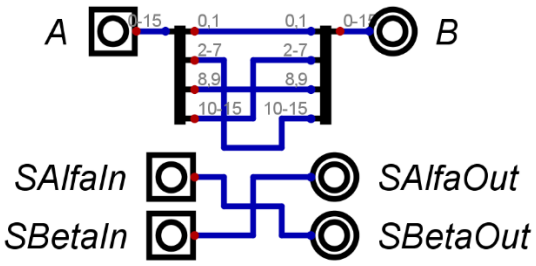


Ilustración 45. Representación de la compuerta NOT en Digital.

La siguiente compuerta cuántica que se diseñará es la compuerta CNOT, en la que se tienen varios escenarios. El primero sería cuando el bit de control es el bit cuántico más significativo; el segundo seria cuando el bit de control es el bit cuántico menos significativo. De manera que se estará manejando esto con un pulso, controlando el paso de información con un multiplexor. Ahora en el primer escenario se tiene que los escalares que cambian de posición son los escalares de los estados $|10\rangle$ y $|11\rangle$, con todo y con su signo claramente. En el segundo escenario los escalares que cambian de posiciones son los escalare que acompañan a los estados $|01\rangle$ y $|11\rangle$, con todo y su signo. Definiendo entonces una entrada de 44 bits para los dos bits cuánticos y una entrada para el control del bit de control, valga la redundancia, se puede entonces diseñar el circuito de la siguiente manera.

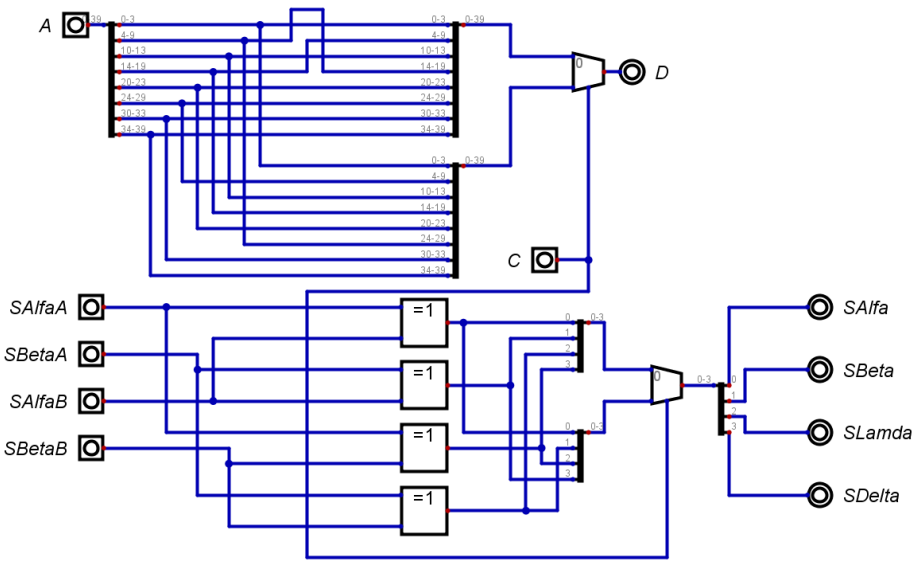


Ilustración 46. Representación de la compuerta CNOT con entrada para dos bits cuánticos unificados.

Fuente de la imagen: Elaboración propia.

Sin embargo, se quiere que la entrada de la compuerta cuántica sean los dos bits que se estará utilizando, ósea tener dos entradas para bits cuánticos. Para esto entonces se tendrá que cambiar la entrada A qué se ve en la ilustración 42. Lo que se hará es intercambiar esta entrada por el circuito de dos bits cuánticos que se describe en la sección 5.1.2. Aplicando esto, entonces, quedara el siguiente circuito para la representación de la compuerta cuántica CNOT.

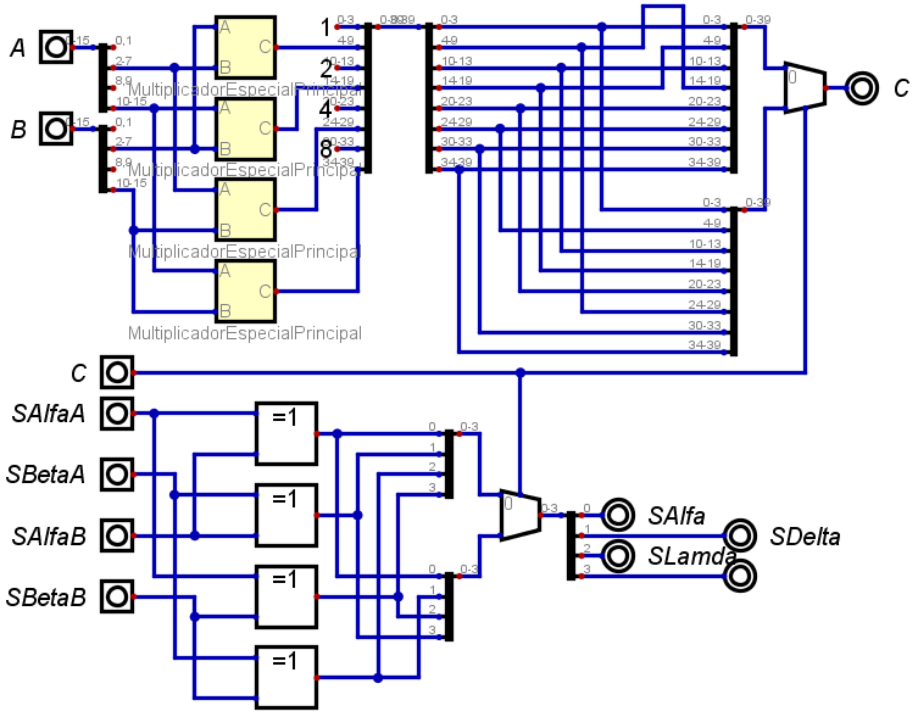


Ilustración 47. Representación de la compuerta CNOT con entrada para dos bits cuánticos por separado.

Fuente de la imagen: Elaboración propia.

Ahora se estará haciendo el diseño para las dos aplicaciones para tres bits cuánticos de la compuerta CNOT. Se aplicarán dos casos distintos para esta compuerta cuántica. El primer caso es cuando el bit cuántico de control es más significativo y el segundo caso es cuando el segundo bit cuántico más significativo es el de control. Por lo que se quiere plasmar los dos casos en un mismo circuito, como se hizo con el control de bit cuántico en el mismo CNOT,

pero de dos bits cuánticos. Ahora debido a que no se quiere saturar el FPGA con escenarios que no son parte de una función aplicable, porque no se ocupa el bit menos significativo sea un bit de control solo un bit que se pueda cambiar, pero que luego se pueda considerar como que no cambio de manera de controlar la respuesta del otro bit cuántico. Ósea manipular el segundo pero que luego la superposición me permita considerar que quien en realidad cambio fue el otro bit cuántico. Debido a esto no se aplicará un caso en el que el bit menos significativo sea el bit de control sobre los otros dos. Dicho esto, y aplicando el análisis realizado para esta aplicación de la compuerta se puede realizar el diseño del circuito equivalente de la siguiente manera.

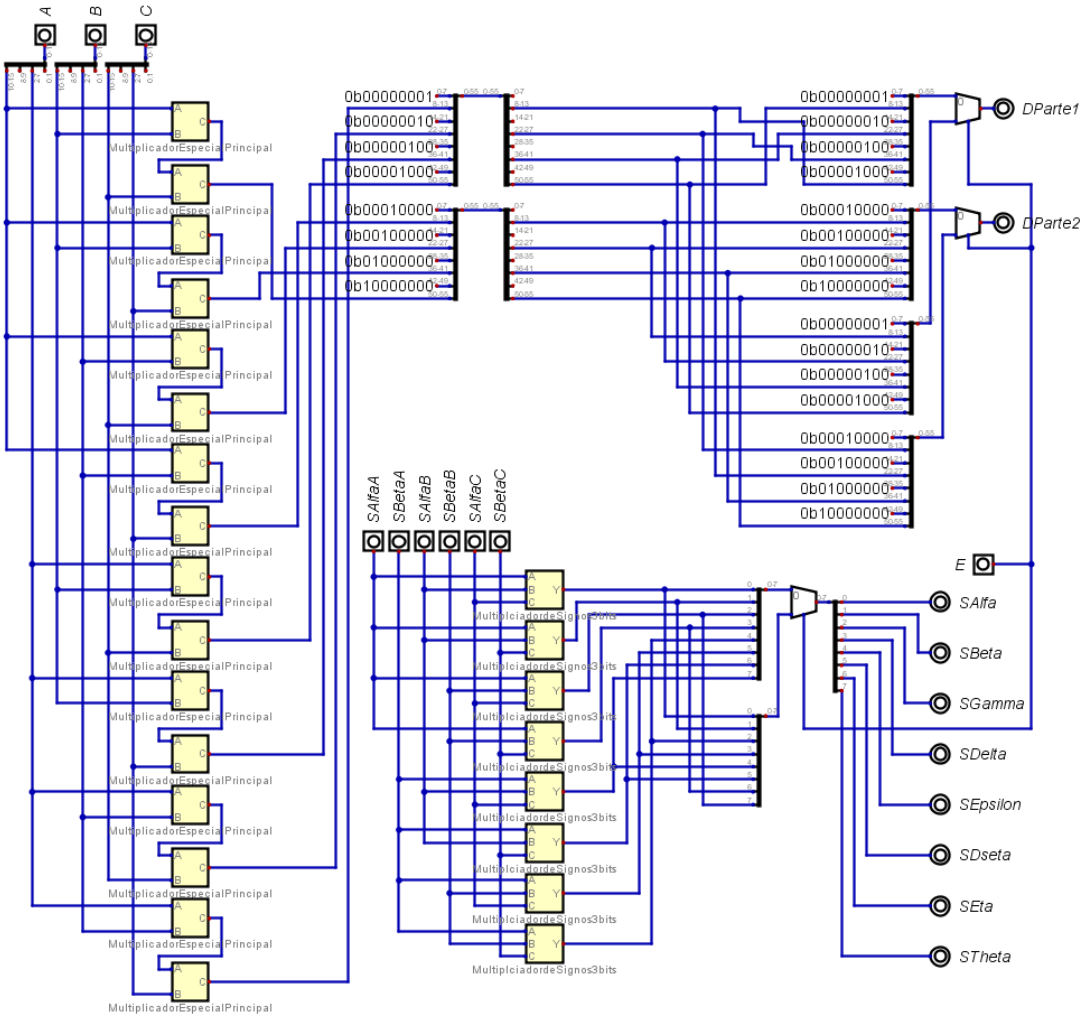


Ilustración 48. Circuito equivalente de la compuerta CNOT aplicada a un control de dos bits cuánticos debido a otro bit, en un sistema de tres bits cuánticos.

Fuente de la imagen: Elaboración propia.

Donde sí la entrada E es 0 entonces el bit de control es el más significativo y si esta entrada es 1 el bit de control es el segundo bit más significativo. Se estará ahora entonces viendo el diseño bajo las condiciones

Por último, se estará haciendo el diseño de la compuerta cuántica ContraCNOT, que como se sabe funciona similar a la compuerta CNOT solo con la pequeña diferencia el cambio se realiza cuando el bit de control es 0. Se utiliza la misma idea que con la compuerta CNOT, tiene la necesidad de indicarle cual es el bit de control, por lo que igual que a la compuerta CNOT se le asignará una entrada de un simple pulso que controlar el flujo de información debido a un multiplexor. En el primer escenario, que es cuando el bit cuántico más significativo es el de control, como ya se sabe, debido al análisis realizado en la sección anterior, los escalares que cambian de posición son los escalares que acompañan a los estados $|00\rangle$ y $|01\rangle$. En el segundo escenario, que es cuando el bit cuántico de control es el menos significativo, los escalares que cambian de posición son los escalares de los estados $|00\rangle$ y $|10\rangle$. A esta analogía se debe agregar que también, al igual que en la compuerta cuántica CNOT, se quiere tener una entrada para cada uno de los bits cuánticos. De manera entonces que el esquema del circuito digital para la representación de la compuerta cuántica ContraCNOT es el siguiente.

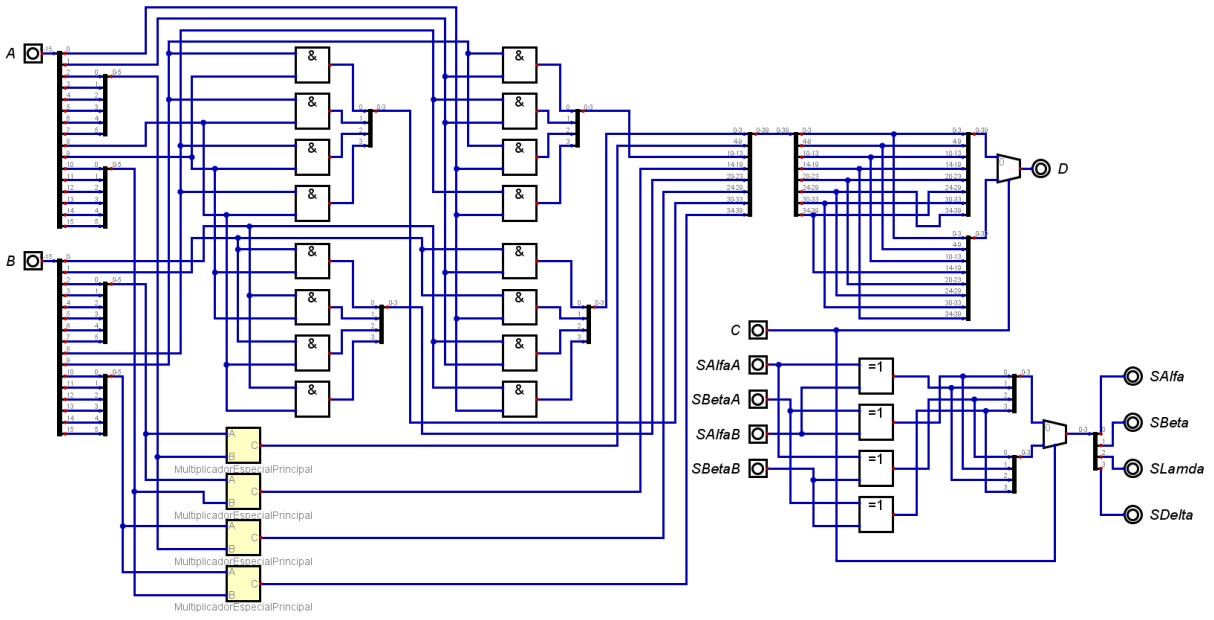


Ilustración 49. Representación de la compuerta ContraCNOT con entrada para dos bits cuánticos por separado.

Fuente de la imagen: Elaboración propia.

Para esta también se tendrá una aplicación de tres bits cuánticos bajo la misma lógica de la aplicación en la compuerta CNOT. Buscando que el más significativo y el que le siga puedan ser bits de control debido a una entrada. Aplicando entonces el análisis realizado en la sección anterior para esta compuerta y tomando en cuenta las mismas condiciones planteadas en la compuerta anterior, se obtiene el siguiente circuito.

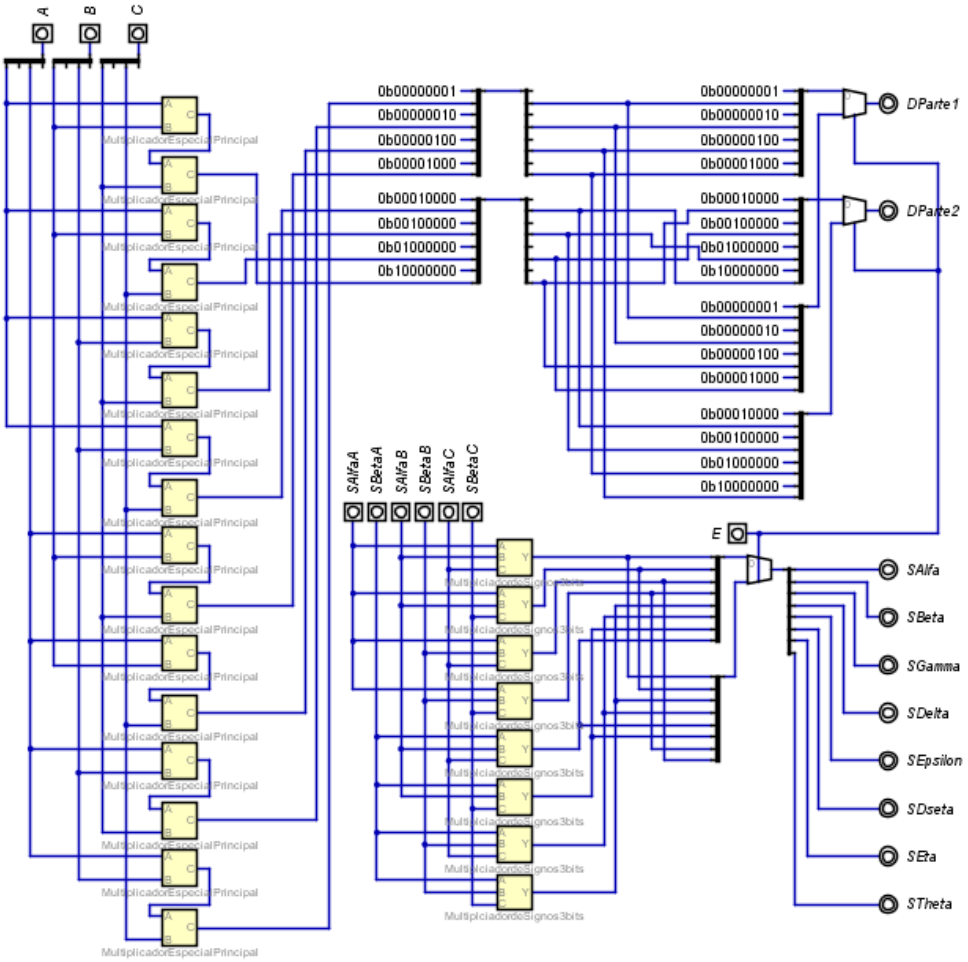


Ilustración 50. Circuito equivalente de la compuerta ContraCNOT aplicada a un control de dos bits cuánticos debido a otro bit, en un sistema de tres bits cuánticos.

Fuente de la imagen: Elaboración propia.

5.2.3 Generar Códigos para los Diseños

En esta sección se generarán los códigos necesarios en Verilog para la generación del esquemático y el archivo .bin necesario para poder cargar al FPGA. Esto se hará por medio del software Digital que ayuda a exportar archivos a Verilog y corriendo este código en el software ISE Design Suite.

Se va a comenzar a generar los códigos de la compuerta Haddamard. Para esto se correrá el archivo en el que se creó el código y en el mismo software Digital se va a utilizar la función exporta a Verilog.

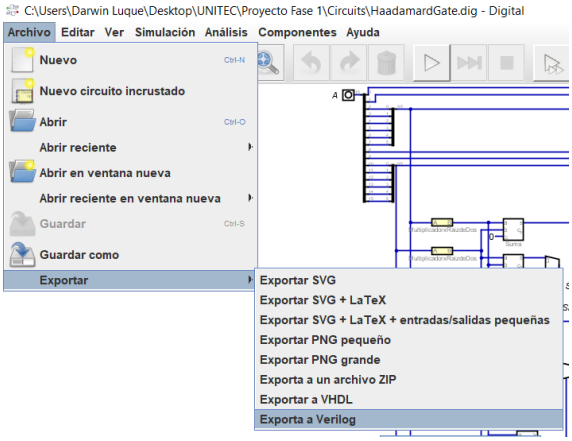


Ilustración 51. Función Exporta a Verilog del software Digital.

Fuente de la imagen: Elaboración propia.

Esta función generara un archivo de tipo V, que es el tipo de archivo para código en Verilog. Este archivo se corre en el software ISE Design Suite. Pero primero se debe crear un proyecto en el software ISE Design Suite. Una vez creado el proyecto se procede a correr la función Implement Top Module de manera de poder revisar la síntesis, la sintaxis y la implementación del diseño.

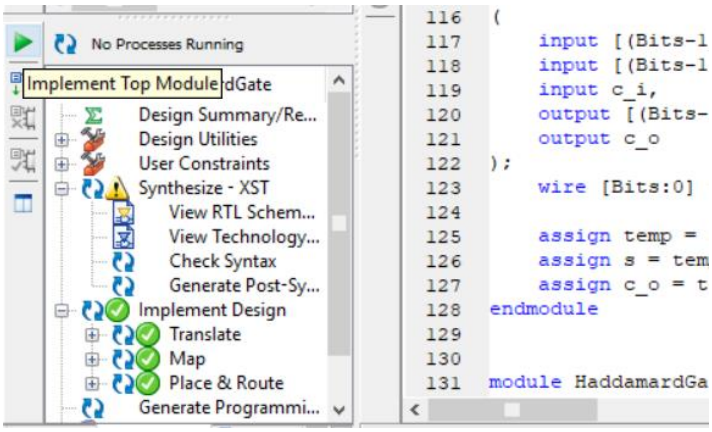


Ilustración 52. Función Implement Top Module del software ISE Design Suite.

Fuente de la imagen: Elaboración propia.

Al ver que la síntesis genera el esquemático, que la sintaxis es correcta y que el diseño es implementable se procede a generar el archivo necesario para la asignación de pines. Para la

comprobación de la funcionalidad de la compuerta cuántica se modificó el circuito de la siguiente manera.

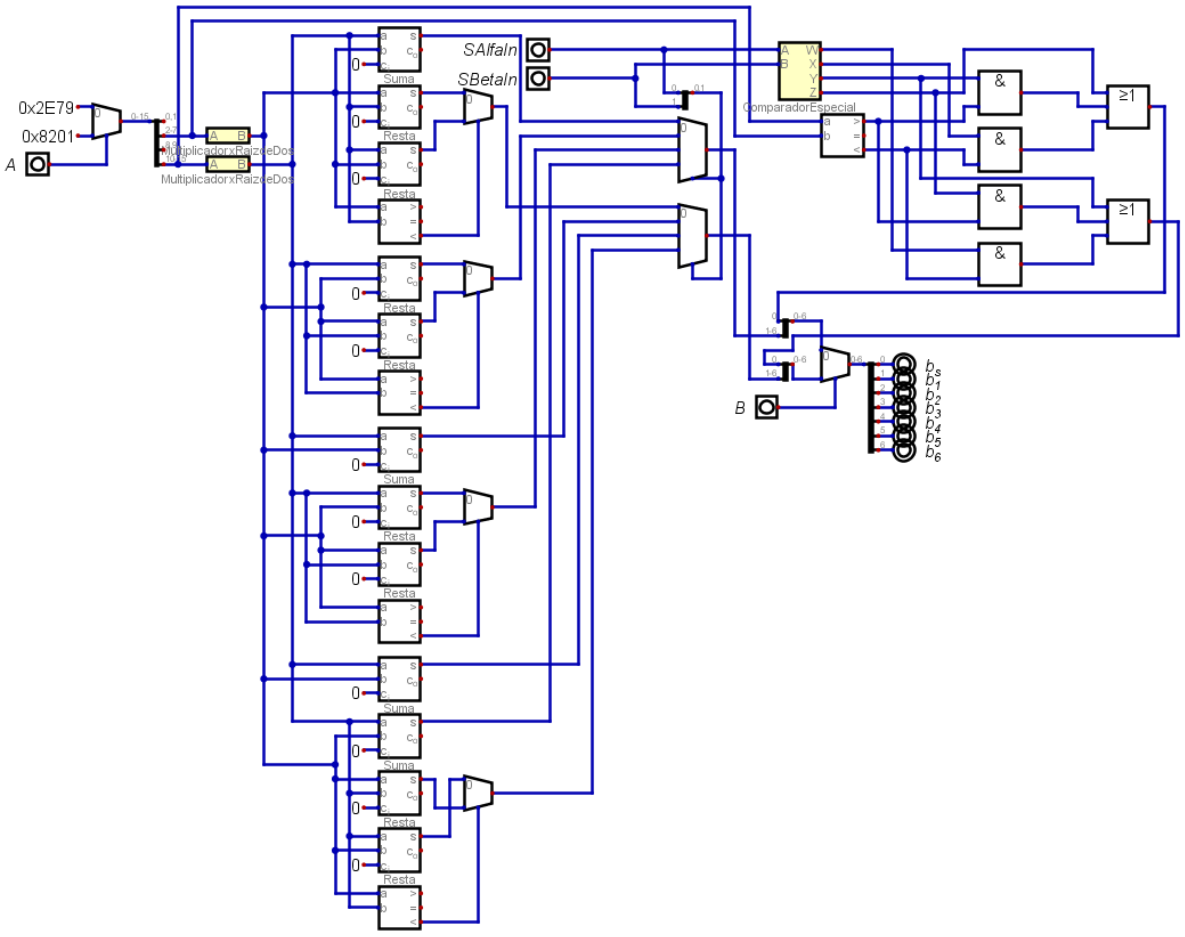


Ilustración 53. Circuito equivalente para compuerta de Haddamard modificado.

Fuente de la imagen: Elaboración propia.

El circuito se modificó de manera que se pueda controlar dos entradas asignadas, en este caso se va a manejar los sets de bits 0b0011011001111001 y 0b1000001000000001, estos son los valores hexadecimales que se ven al principio del primer multiplexor. Se tomaron simplemente dos casos distintos para luego ver los resultados matemáticos de estos y compararlo con el resultado devuelto por la FPGA. Estos son debido al mapeo realizado a los siguientes bits cuánticos.

$$|x\rangle = \frac{1}{3}|0\rangle + \frac{2\sqrt{2}}{3}|1\rangle, \qquad |y\rangle = |0\rangle$$

Donde el bit cuántico x es el primer set de bits y el bit cuántico y es el segundo set de bits, tomando en cuenta solo las magnitudes los signos se pusieron predeterminadamente

positivos. Sin embargo, los signos tendrán entradas individuales de manera de poder controlarlos y poder observar más claro el efecto que tienen estos sobre la compuerta. Se va a estar midiendo la funcionalidad de las compuertas cuánticas por medio de las salidas $b_6, \dots, b_1, b_s$, los cuales devuelven los valores de los escalares. Para ahorrar recursos y poder utilizar los componentes disponibles en la Mimas V2 se controla la información vista en estas salidas por medio de un multiplexor. El cual con el pulso generado por la entrada B va dejar pasar los escalares dependiendo de su valor. Si esta variable B es 0 o 1 devuelve el escalar del estado $|0\rangle$ o $|1\rangle$, respectivamente. Estas se les asignará puertos en la Mimas V2 de manera de que por medio de la FPGA se controlen las LEDs disponibles en esta misma placa.

Sabiendo esto entonces se procede a crear el archivo que va a definir las variables en el FPGA. Este archivo es de tipo UCF. Para saber los pines que se estarán asignando se tiene que plantear que necesidad tiene el circuito. Por ejemplo, los pines de entrada tienen que ser componentes electrónicos que no necesiten retención. Ya que no se tiene programada esta función. Para eso entonces se estará haciendo uso del dip switch disponible en la placa. Por lo que se ocupara cuatro pines de entrada de este dip switch, para asignar los pines se tiene que ver el datasheet de la Mimas V2, este datasheet estará disponible en la sección de Anexos. Según el datasheet los primeros cuatro puertos del dip switch están en los puertos F17, F18, E16 y E18, respectivamente. Otro asunto a considerar es que los dip switches y botones disponibles en la Mimas V2 no tienen resistencias de pull up por lo que se tendrá que declarar este concepto en el código. Sin embargo, a función pull up va a causar que las entradas actúen con lógica inversa, esto debido a que el pulso lo que va a causar es que la entrada se vaya a tierra, causando que al tener el pulso puesto la lectura lógica sea 0. Es por esto que en el circuito modificado de la compuerta de Haddamard, y en el resto de circuitos que se tiene por diseñar, las entradas van a tener un NOT clásico puestos antes del multiplexor u otro operador que se quiera controlar con el mismo.

Las salidas se estarán representando con las luces LEDs disponibles en la Mimas V2. Según el datasheet las luces LEDs, de las cuales no sobrara una ya que la placa cuenta con ocho de estas, están en los pines P15, P16, N15, N16, U17, U18, T17 y T18. Ahora estos están colocados en orden del primer LED al último de izquierda a derecha, por lo que se busca ver lo más a la izquierda posible al signo del escalar por lo que se le asigna el primer LED y los bits de salida, ósea los que devuelven el valor del escalar mapeado según la tabla 9 a binario,

se asignará el bit más significativo al primer LED, posterior al ya asignado al signo, y el bit menos significativo, al último LED. Esto de manera de leer el numero en la placa de izquierda a derecha con normalidad. Planteado esto se puede asignar los siguientes pines a las variables del circuito modificado para la verificación de la funcionalidad de la compuerta Haddamard.

```
1  NET "A" PULLUP;
2  NET "A" LOC = F17;
3  NET "SAlfaIn" PULLUP;
4  NET "SAlfaIn" LOC = F18;
5  NET "SBetaIn" PULLUP;
6  NET "SBetaIn" LOC = E16;
7  NET "B" PULLUP;
8  NET "B" LOC = E18;
9  NET "b_s" LOC = P15;
10 NET "b_1" LOC = T17;
11 NET "b_2" LOC = U18;
12 NET "b_3" LOC = U17;
13 NET "b_4" LOC = N16;
14 NET "b_5" LOC = N15;
15 NET "b_6" LOC = P16;
```

Ilustración 54. Asignación de pines a las variables del circuito de la compuerta de Haddamard.

Fuente de la imagen: Elaboración propia.

Ahora para la siguiente compuerta, que será la NOT, se realizará las mismas funciones del inicio de la compuerta de Haddamard. Primero, se modificará el circuito, para medir la funcionalidad del circuito. Una vez realizado esto se exportará el archivo a Verilog como se explica con la compuerta anterior, y luego se correrá la función Implement Top Module. Para poder seguir con la siguiente etapa de pines, primero se planteará la modificación que se le hizo al circuito.

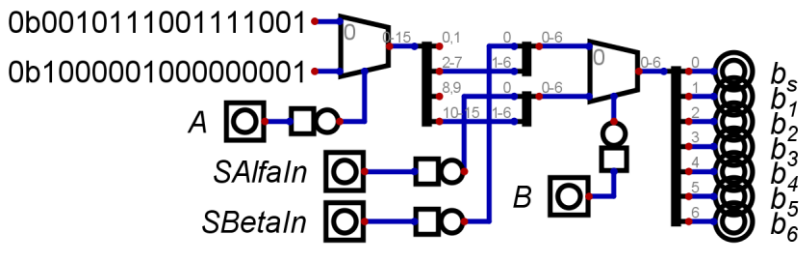


Ilustración 55. Circuito equivalente para la compuerta NOT modificado.

Fuente de la imagen: Elaboración propia.

El circuito se modificó de la misma manera que se modifica el circuito para la compuerta de Haddamard, en cuánto al control de las entradas. Solo que, claro, este circuito funciona de tal manera que funcionaria la compuerta cuántica NOT. Se asignó los mismos sets de bits para la entrada del bit cuántico con el mismo pulso de control. Y a las salidas se les denomino con la misma nomenclatura. De esta manera la declaración de variables es la misma que en el caso de la compuerta de Haddamard. Por ende, el archivo UCF que se estará utilizando es el mismo que para la compuerta de Haddamard.

Ahora se estará observando la compuerta CNOT. De manera que el procedimiento para la generación y verificación del código es el mismo que las compuertas anteriores. Se tendrá que modificar el circuito, luego exportar el archivo desde el software digital a Verilog, después correr el archivo de Verilog en el ISE Design Suite, luego se corre la función Implement Top Module, para ver si la síntesis, sintaxis e implementación son correctos y finalmente generar el archivo para los pines. Entonces la modificación necesaria para la verificación de la funcionalidad de la equivalencia de la compuerta cuántica CNOT es la siguiente.

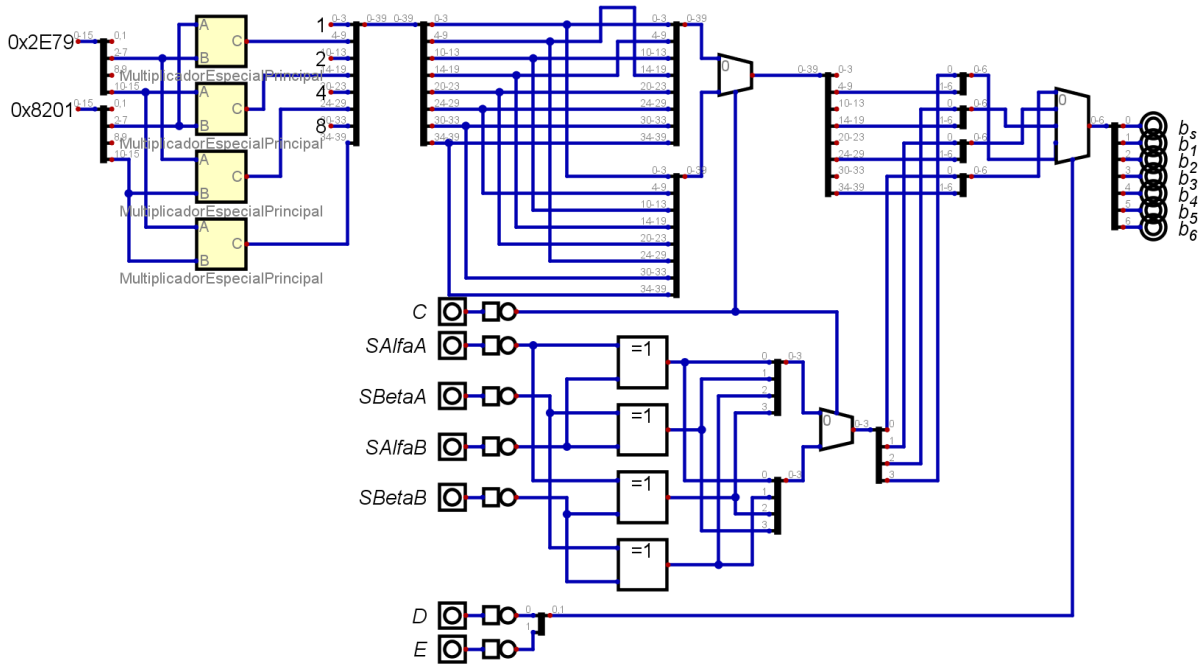


Ilustración 56. Circuito equivalente de la compuerta CNOT modificado.

Fuente de la imagen: Elaboración propia.

En este caso del CNOT se estarán usando entradas para los bits cuánticos constantes de manera de ahorrar recursos, además que la prueba de estas es suficiente para la verificación

de su funcionalidad. Los bits cuánticos siendo representados son los mismo que ya se plantearon en las compuertas anteriores. Siendo asignado uno de esos dos bits para cada entrada. En este circuito se va a poder controlar la salida con el multiplexor, con la misma idea que con las compuertas anteriores. Sin embargo, ya que esta compuerta tiene una salida en base a dos bits cuánticos se manejarán cuatro valores de escalares por lo que se ocupará dos entradas para el control de información en el multiplexor de las salidas, entradas que se llaman D y E. Ahora también para este circuito se ocupará una entrada para el control del bit cuántico de control, valga la redundancia. Y finalmente, una entrada para cada signo de entrada de ambos bits cuánticos por separado. Por lo que se estará utilizando el dip switch disponible en la Mimas V2 para todas estas entradas, por lo que se tendrá que ver los pines disponibles en el datasheet. Para C, SAlfaA, SBetaA, SAlfaB, SBetaB, D y E se estará usando los puertos F17, F18, E16, E18 y D18, respectivamente. Las salidas al ser igual a las compuertas anteriores se le asignaran los mismos puertos de los LEDs disponibles en la Mimas V2.

```
1  NET "C" PULLUP;
2  NET "C" LOC = F17;
3  NET "SAlfaA" PULLUP;
4  NET "SAlfaA" LOC = F18;
5  NET "SBetaA" PULLUP;
6  NET "SBetaA" LOC = E16;
7  NET "SAlfaB" PULLUP;
8  NET "SAlfaB" LOC = E18;
9  NET "SBetaB" PULLUP;
10 NET "SBetaB" LOC = D18;
11 NET "D" PULLUP;
12 NET "D" LOC = D17;
13 NET "E" PULLUP;
14 NET "E" LOC = C18;
15 NET "b_s" LOC = P15;
16 NET "b_1" LOC = T17;
17 NET "b_2" LOC = U18;
18 NET "b_3" LOC = U17;
19 NET "b_4" LOC = N16;
20 NET "b_5" LOC = N15;
21 NET "b_6" LOC = P16;
```

Ilustración 57. Asignación de pines para las variables de la compuerta CNOT.

Fuente de la imagen: Elaboración propia.

Ahora se tiene que generar el código de la compuerta CNOT aplicada al caso de los tres bits cuántico. Los pasos para la generación del código es siempre exportar a Verilog desde el software Digital. Generado el código en Verilog se crea un proyecto en el software ISE

Design Suite en el que se corre el código. De manera entonces de poner en función la función, valga la redundancia, de Implement Top Module. Verificado entonces la implementación, la sintaxis y síntesis del código. Una vez verificado el código se procede a crear el archivo UCF para la asignación de pines del circuito. Ahora primero se tiene que hacer una pequeña modificación al circuito, para lo que se generará un archivo nuevo con la misma lógica, pero agregando nuevas entradas y salidas de manera de poder tener un parámetro para medir su funcionalidad. De manera entonces que el circuito modificado se ve de la siguiente manera.

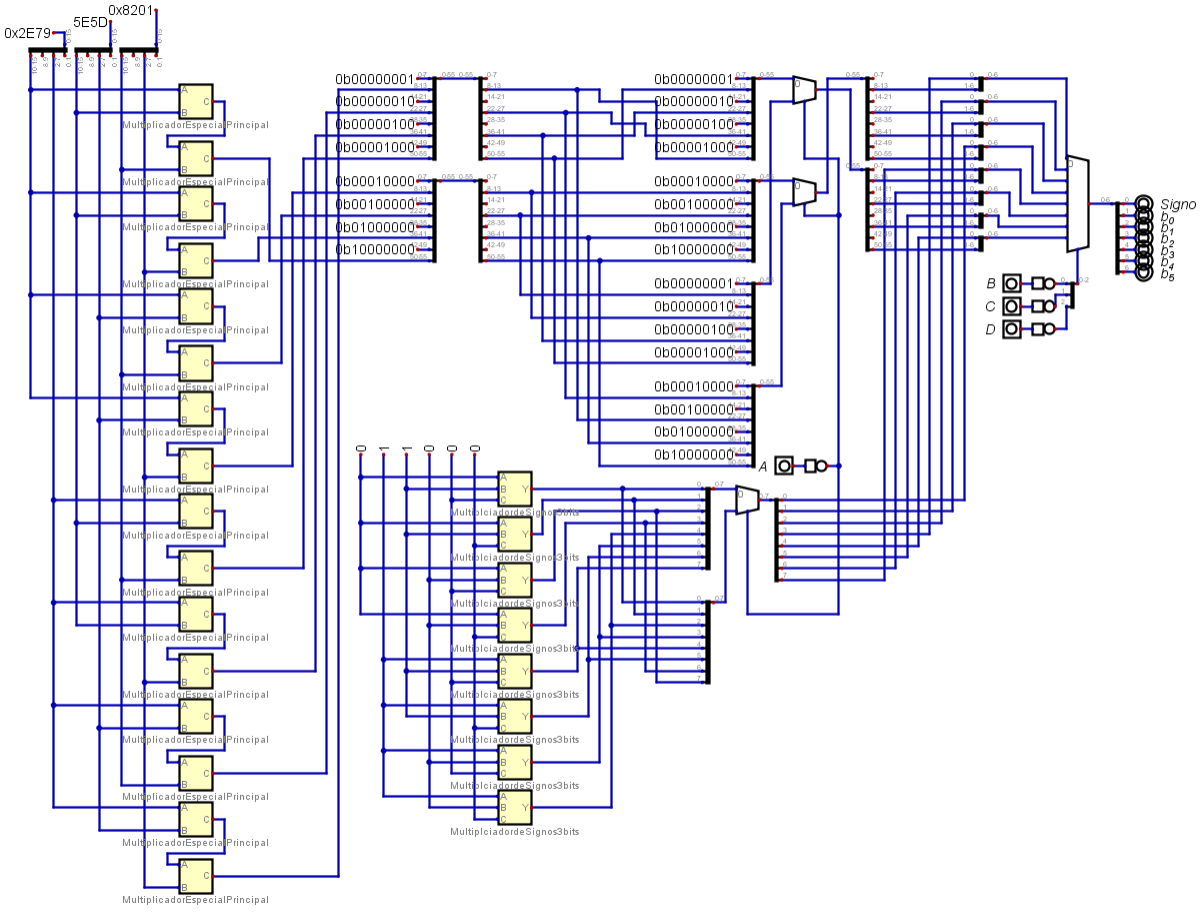


Ilustración 58. Circuito para la compuerta CNOT con la aplicación a tres bits cuánticos modificada.

Fuente de la imagen: Elaboración propia.

El circuito se dejó con entradas y signos constantes, en este circuito también se tuvo que agregar una constante más. Esta nueva constante es el set de bits 0b0101111001011101. Considerando entonces los bits para los signos y la nueva constante los bits cuánticos siendo

representados en la entrada del circuito modificado para la compuerta CNOT aplicada a tres bits cuánticos son los siguientes.

$$|x\rangle = \frac{1}{3}|0\rangle - \frac{2\sqrt{2}}{3}|1\rangle, \qquad |z\rangle = -\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle, \qquad |y\rangle = |0\rangle$$

Donde x es el bit más significativo, y es el segundo bit más significativo y z es el menos significativo. Estas entradas entonces, incluyendo el bit para elegir el bit cuántico de control, serán simuladas en la Mimas V2 por el dip switch, donde se le asignaran los primeros 4 switches de estos, pines ya descritos anteriormente y que se verá específicamente en el archivo UCF. Las salidas serán asignadas a las luces LEDs disponibles en la Mimas V2, salidas también serán descritas en el archivo UCF. De tal manera entonces que el archivo UCF se vera de la siguiente manera.

```
1  NET "A" PULLUP;
2  NET "A" LOC = F17;
3  NET "B" PULLUP;
4  NET "B" LOC = F18;
5  NET "C" PULLUP;
6  NET "C" LOC = E16;
7  NET "D" PULLUP;
8  NET "D" LOC = E18;
9  NET "Signo" LOC = P15;
10 NET "b_0" LOC = T17;
11 NET "b_1" LOC = U18;
12 NET "b_2" LOC = U17;
13 NET "b_3" LOC = N16;
14 NET "b_4" LOC = N15;
15 NET "b_5" LOC = P16;
```

Ilustración 59. Asignación de variables para la compuerta CNOT con la aplicación a los tres bits cuánticos.

Fuente de la imagen: Elaboración propia.

Para la última compuerta cuántica, la ContraCNOT. El procedimiento a seguir es el mismo, buscando siempre modificar el circuito de manera de poder conseguir una forma en la que se pueda verificar la funcionalidad del circuito, luego se exporta el diseño a Verilog, luego se corre el archivo en ISE Design Suite, hecho esto se va a correr la función Implement Top Module para verificar la implementación, síntesis y sintaxis del código y finalmente generar el archivo UCF para la asignación de pines a cada variable. De manera entonces que la modificación al circuito se ve de la siguiente manera.

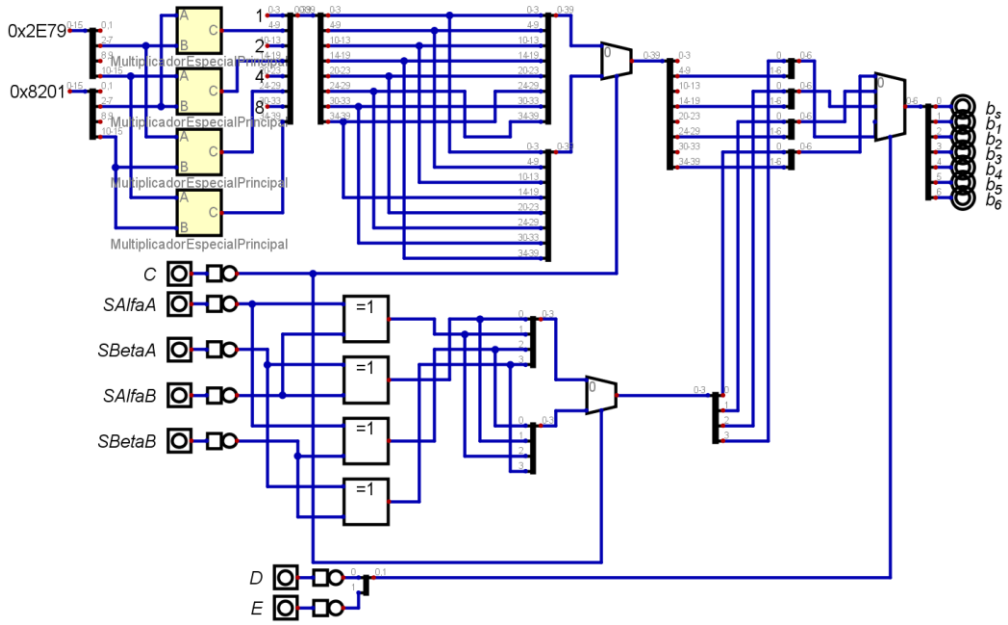


Ilustración 60. Circuito equivalente para la compuerta ContraCNOT modificado.

Fuente de la imagen: Elaboración propia.

Las modificaciones se realizaron pensando siempre en la misma lógica que la compuerta CNOT. Las variables a utilizar son las mismas que los de la compuerta CNOT por lo que la declaración de variables es la misma. Con la diferencia claro del funcionamiento y el resultado que estas salidas van a tener debido a las entradas.

Por ultimo en cuanto a la generación de códigos se tiene la aplicación a tres bits cuánticos de la compuerta ContraCNOT. Esta se rige bajo la misma idea de los casos del bit de control. De tal manera entonces que el circuito se ve de la siguiente manera.

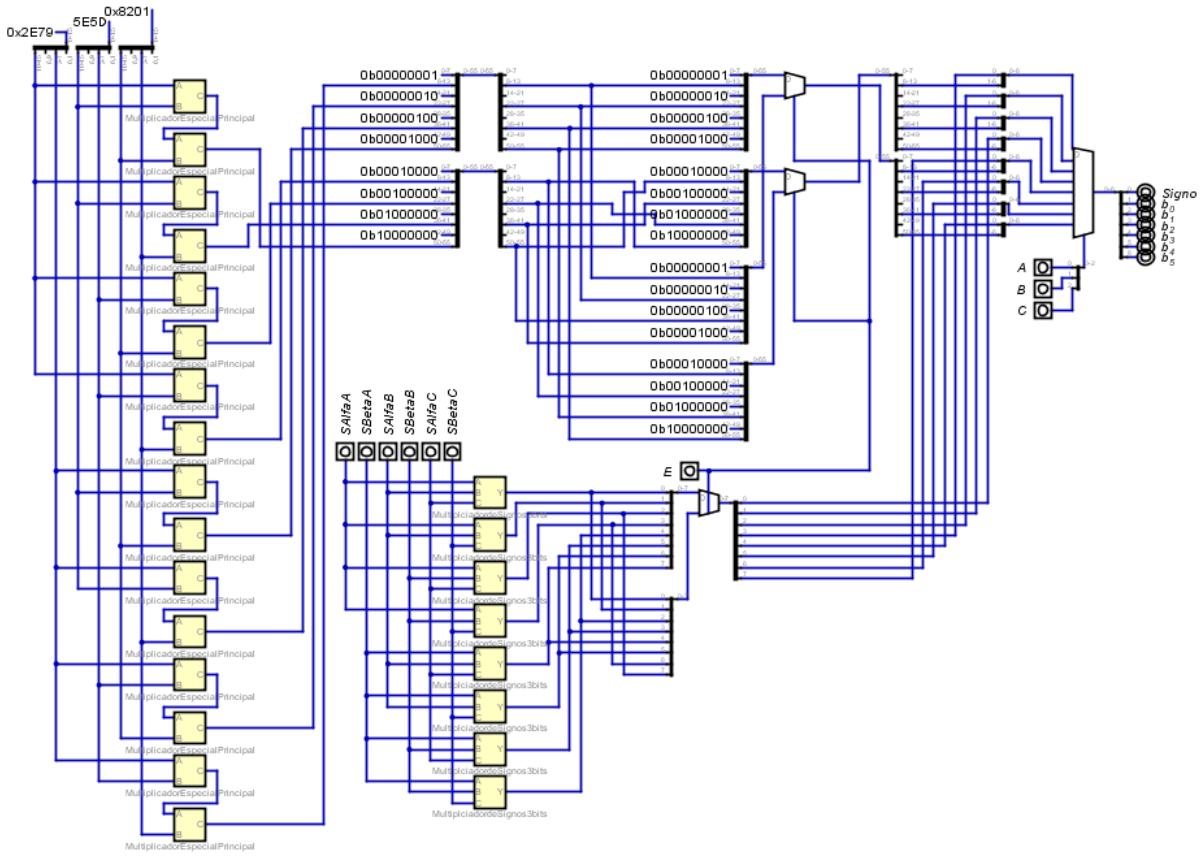


Ilustración 61. Circuito equivalente de la compuerta ContraCNOT con la aplicación de los tres bits cuánticos modificada.

Fuente de la imagen: Elaboración propia.

Como se puede observar, debido a la falta de recursos, se aplicó el mismo concepto de manejar la entrada de bits cuánticos constante. Manejar un solo set de bits que son controlados por medio del multiplexor que deja pasar uno u otro escalar debido al set de bits de control del multiplexor A, B y C. Debido a que se manejan las misma entradas y salidas que la compuerta cuántica CNOT bajo la misma aplicación se estará utilizando el mismo archivo UCF.

5.2.4 Verificación de la Funcionalidad de los Circuitos Diseñados

En la presente sección se estará verificando las funciones de las compuertas cuánticas descritas en este ciclo por lo que se aplicarán los códigos realizados en la sección anterior, códigos que se estarán subiendo al FPGA. Sin embargo, no se han generado el archivo el cual se utiliza para programar la FPGA, este archivo es de tipo BIN. Este archivo se obtendrá de cualquier proyecto creado en ISE Design Suite. Para generar el archivo simplemente se

tiene que utilizar la función Generate Programming File, en el cual cabe destacar que sin un archivo UCF no se puede generar.

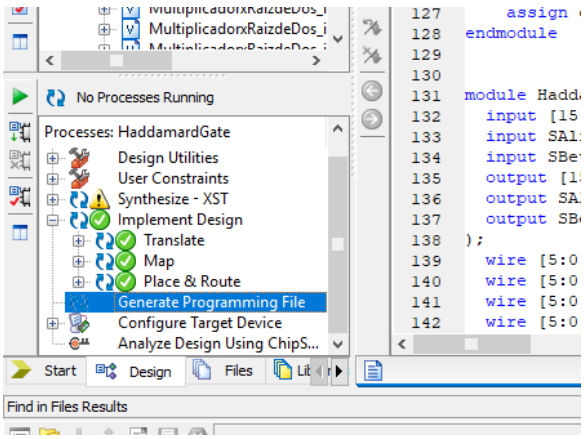


Ilustración 62. Función Generate Programming File en el software ISE Design Suite.

Fuente de la imagen: Elaboración propia.

Sin embargo, un pequeño detalle importante que se debe considerar es que esta función, que genera 3 tipos de archivos distinto, trae predeterminadamente la opción de generar el archivo BIN apagado. Por lo cual simplemente se tiene que ir a las propiedades de la función y activar la generación del archivo BIN. Generado el archivo BIN para el proyecto que se quiera subir a la FPGA se procede a hacer pues esto mismo, subir el código al FPGA.

Para esto se usará el software Mimas Board Configuration. Para la subida simplemente se tiene que seleccionar el puerto en el que se tiene la FPGA y luego seleccionar el archivo BIN que se quiere subir al FGPA y presionar la función Program Board. Solo se deja entonces que el programa suba la programación al FPGA y listo, (*Numato Lab*, s. f.).

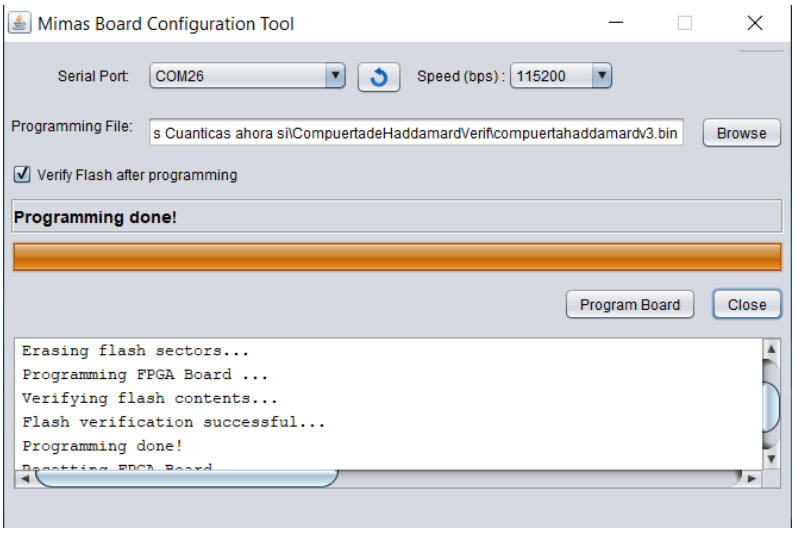


Ilustración 63. Configuración para la programación del FPGA por medio del software Mimas Configuration Tool.

Fuente de la imagen: Elaboración propia.

Para esta ilustración se tomó como ejemplo la compuerta de Haddamard. Se repetirá estos procesos descritos para cada una de las compuertas y para ya los algoritmos.

Sabiendo esto entonces se procede a programar en la FPGA la compuerta de Haddamard de manera entonces de ver cómo reacciona el FPGA y si las respuestas que presenta son tales como los parámetros matemáticos de la compuerta de Haddamard explicados en la teoría de sustento del capítulo 3. Se presentará entonces los resultados que tiene la FPGA y se observará si la respuesta de este da tal como debería de dar. Teniendo en cuenta la asignación de las variables descritas en la sección anterior, se estará usando los nombres de las variables.

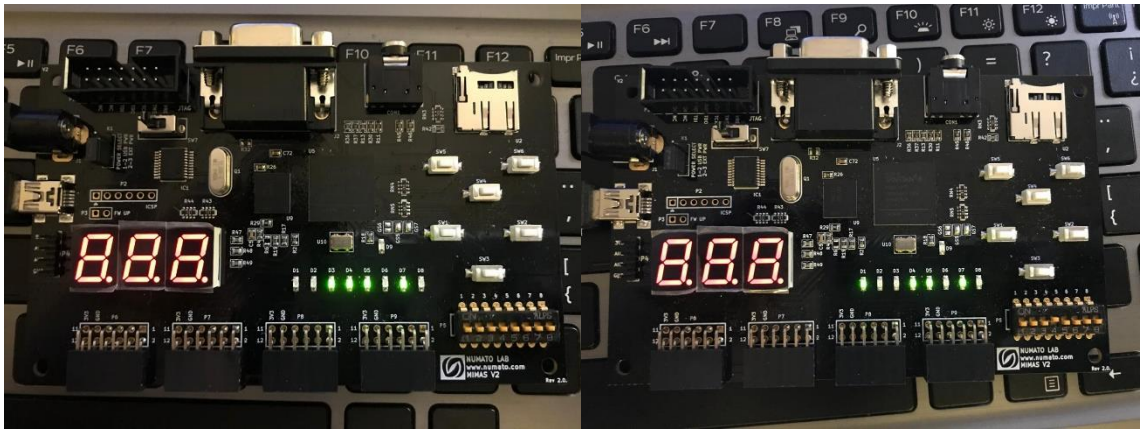


Ilustración 64. FGPA configurado con la compuerta de Haddamard: caso 1.

Fuente de la imagen: Elaboración propia.

Como se puede observar el bit cuántico debido a que A es 0 es el bit representado por el set de bits 0b0011011001111001, y también se observa que los bits de los signos son ambos 0 por lo que entonces ambos escalares de entrada son positivos. Esto ocurre en las dos imágenes de la ilustración 64. La diferencia recae en la variable B en la imagen de la derecha se ve que este bit es 1 y en la izquierda es 0. Si este bit es 0 las LEDs devuelven el valor de α y si es 1 estas devuelven el valor de β. Tomando en cuenta entonces el bit cuántico siendo representado, este al pasar por la compuerta cuántica de Haddamard se ve afectado matemáticamente tal que.

$$H|x\rangle = \frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix}\begin{pmatrix}\frac{1}{3}\\ \frac{2\sqrt{2}}{3}\end{pmatrix} = \begin{pmatrix}\frac{1+2\sqrt{2}}{3\sqrt{2}}\\ \frac{1-2\sqrt{2}}{3\sqrt{2}}\end{pmatrix} \approx \begin{pmatrix}0.902\\ -0.431\end{pmatrix} = 0.902|0\rangle - 0.431|1\rangle$$

Teniendo estos valores se mapearán los sets de bits necesarios para la representación del escalar en base a la tabla 9. De tal forma entonces que el set de bits se ve de la siguiente manera.

$$\alpha = +0.902 \rightarrow 0011101, \qquad \beta = -0.431 \rightarrow 1001101$$

Valores que al comparar con las luces LEDs encendidas en la Mimas V2 se observa que tienen el mismo set de bits. Sin embargo, para dar una mayor certeza de la funcionalidad del circuito y la FPGA se planteará un caso más. Donde se usará el segundo set de bits y se aplicará un negativo en los escalares de entrada.

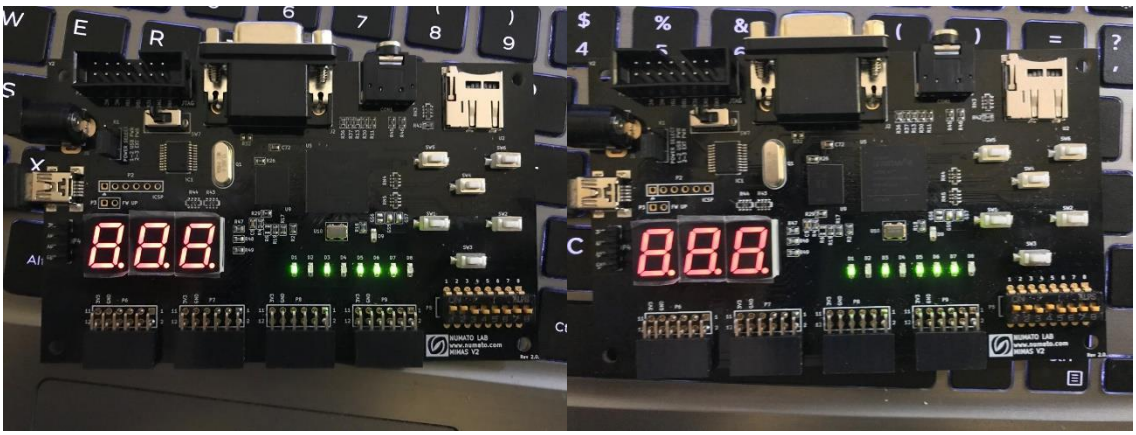


Ilustración 65. FGPA configurado con la compuerta de Haddamard: caso 2.

Fuente de la imagen: Elaboración propia.

Como se puede observar en este caso A tiene valor de 1 por lo que entonces se está usando el set de bits 0b1000001000000001 y también se tiene activado el bit para el signo de α de entrada y desactivado el bit de β de entrada. Y al igual que en el caso anterior la imagen de la izquierda representa el valor del α de salida y a la derecha el valor de β de salida. Tomando en cuenta entonces el bit cuántico siendo representado, este al pasar por la compuerta cuántica de Haddamard se ve afectado matemáticamente tal que.

$$H|y\rangle = \frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix}\begin{pmatrix}-1\\ 0\end{pmatrix} = \begin{pmatrix}-\frac{1}{\sqrt{2}}\\ -\frac{1}{\sqrt{2}}\end{pmatrix} \approx \begin{pmatrix}-0.707\\ -0.707\end{pmatrix} = -0.707|0\rangle - 0.707|1\rangle$$

Teniendo estos valores se mapearán los sets de bits necesarios para la representación del escalar en base a la tabla 9. De tal forma entonces que el set de bits se ve de la siguiente manera.

$$\alpha = -0.707 \rightarrow 1010111, \qquad \beta = -0.707 \rightarrow 1010111$$

Si se observa en la ilustración 65 se ve que ambos valores, respectivamente, concuerdan con lo devuelto por el FPGA. Por lo que con estos dos casos comprobados se puede decir que el diseño del circuito para la representación de la compuerta de Haddamard funciona correctamente.

Ahora se estará probando la compuerta cuántica NOT, por lo que se estará configurando el FPGA con la programación de esta compuerta. Se estará trabajando con el mismo acercamiento, se presentará el FPGA bajo distintos casos y se verá si este concuerda con la teoría de sustento presentada en el capítulo 3.

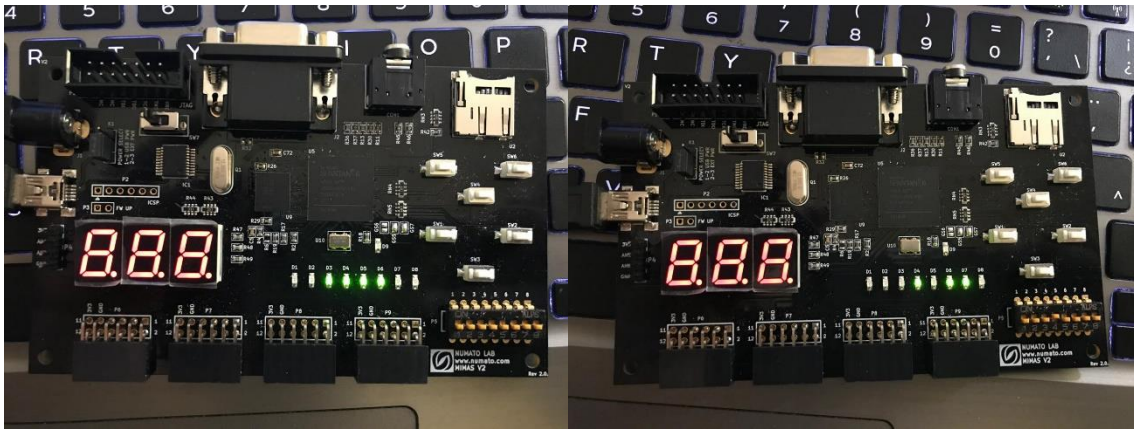


Ilustración 66. FGPA configurado con la compuerta NOT: caso 1.

Fuente de la imagen: Elaboración propia.

Como se puede observar en la ilustración anterior en ambos casos se tiene que la variable A es igual a 0 por lo que entonces se está representando el bit cuántico con el set de bits 0b1000001000000001. La diferencia en entre las dos imágenes es el bit B para la elección del escalar de lectura. En la imagen de la izquierda se puede observar el valor del escalar de α ya que el valor de B es igual 0 y en la imagen de la derecha se puede observar el

valor del escalar β ya que el valor de B es igual a 1. Ahora si se tiene al bit cuántico siendo representado pasar por una compuerta cuántica NOT se obtendría el siguiente resultado matemático.

$$X|x\rangle = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} \frac{1}{3} \\ \frac{2\sqrt{2}}{3} \end{pmatrix} = \begin{pmatrix} \frac{2\sqrt{2}}{3} \\ \frac{1}{3} \end{pmatrix} \approx \begin{pmatrix} 0.942 \\ 0.333 \end{pmatrix} = 0.942|0\rangle + 0.333|1\rangle$$

Teniendo estos valores se mapearán los sets de bits necesarios para la representación del escalar en base a la tabla 9. De tal forma entonces que el set de bits se ve de la siguiente manera.

$$\alpha = 0.942 \rightarrow 0011110, \qquad \beta = 0.333 \rightarrow 0001011$$

Que si se observa en la ilustración 66 se obtienen los mismos valores en esta deducción matemática que en la FPGA por lo que entonces se puede observar que esta funciona tal y como debe. Sin embargo, se puede someterla a un caso más donde se va a utilizar el otro set de bits alternativo aplicando signos de manera de observar si estos se mueven como deberían.

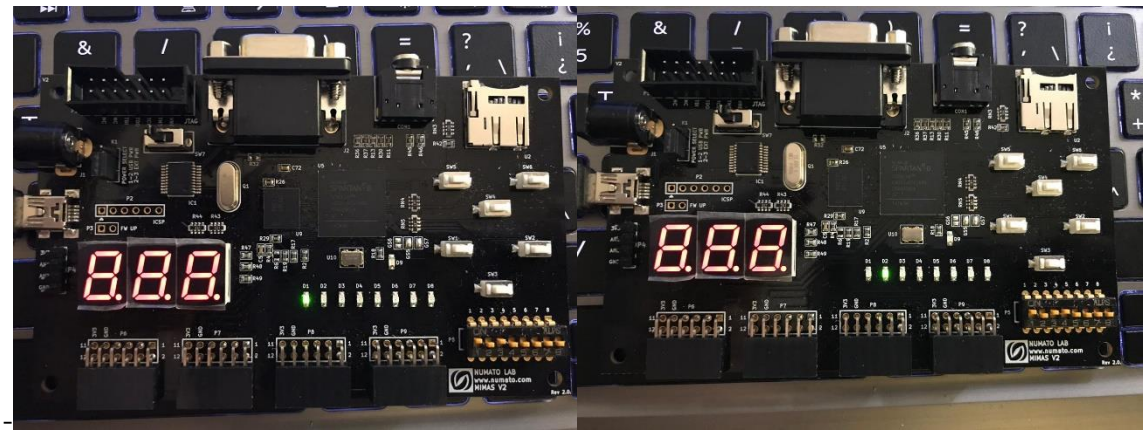


Ilustración 67. FGPA configurado con la compuerta NOT: caso 2.

Fuente de la imagen: Elaboración propia.

Como ya se había mencionado se busca manejar la otra alternativa de representación de un bit cuántico por lo que entonces en ambas imágenes se puede ver que se puso el bit de la variable A en 1. También se activa el bit del signo del escalar β de entrada. Por lo que entonces la representación matemática se ve de la siguiente manera.

$$X|y\rangle = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ -0 \end{pmatrix} = \begin{pmatrix} -0 \\ 1 \end{pmatrix} = -0|0\rangle + |1\rangle$$

Teniendo estos valores se mapearán los sets de bits necesarios para la representación del escalar en base a la tabla 9. De tal forma entonces que el set de bits se ve de la siguiente manera.

$$\alpha = -0 \rightarrow 1000000, \qquad \beta = 1 \rightarrow 0100000$$

Como se puede ver estos valores coinciden con los valores devueltos por la FPGA por lo que entonces esto ayuda a concluir que tanto el diseño del circuito es funcional y representa bien a la compuerta cuántica NOT, y que el FPGA tiene la capacidad de correr y emular este circuito de manera correcta.

Ahora se estará poniendo a prueba la funcionalidad de la compuerta cuántica CNOT por lo que entonces se configurará la FPGA con el archivo BIN generado debido al diseño modificado de esta misma compuerta. Por lo que también se buscara comparar los valores devueltos por el FPGA con los valores de la representación matemática del bit cuántico utilizado.

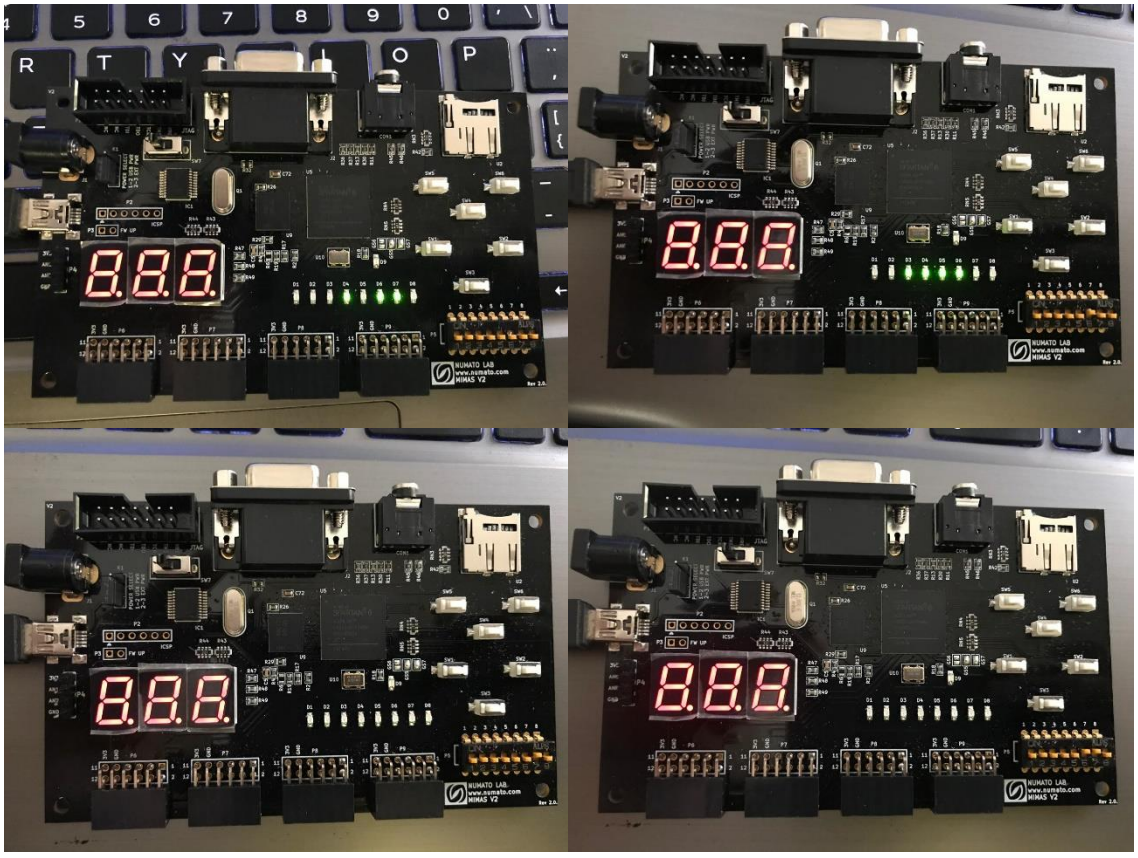


Ilustración 68. FPGA configurado con la compuerta CNOT: caso 1.

Fuente de la imagen: Elaboración propia.

Como se puede observar los cuatro casos presentados tienen como bit de control al bit cuántico más significativo. La diferencia entre ellas es el escalar que representan, este depende de los últimos dos pulsos del dip switch que se definieron como las variable D y E. Por lo que la imagen superior izquierda se ve el escalar de respuesta α, la imagen superior derecha se ve el escalar de respuesta β, la imagen inferior izquierda se ve el escalar de respuesta λ y la imagen inferior derecha se ve el escalar de respuesta δ. Los signos de los cuatro escalares de entrada están en 0 por lo que entonces la representación matemática del presente caso se ve de la siguiente manera.

$$|yx\rangle = \frac{1}{3}|00\rangle + \frac{2\sqrt{2}}{3}|01\rangle \rightarrow CNOT|yx\rangle = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} \frac{1}{3} \\ \frac{2\sqrt{2}}{3} \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} \frac{1}{3} \\ \frac{2\sqrt{2}}{3} \\ 0 \\ 0 \end{pmatrix} \approx \begin{pmatrix} 0.333 \\ 0.942 \\ 0 \\ 0 \end{pmatrix}$$

$$= 0.333|00\rangle + 0.942|01\rangle$$

Teniendo estos valores se mapearán los sets de bits necesarios para la representación del escalar en base a la tabla 9. De tal forma entonces que el set de bits se ve de la siguiente manera.

$$\alpha = 0.333 \rightarrow 0001011, \qquad \beta = 0.942 \rightarrow 0011110, \qquad \lambda = 0 \rightarrow 0000000,$$
$$\delta = 0 \rightarrow 0000000$$

Que como se puede observar en la ilustración 68 estos valores son los que se ven reflejados por los LEDs en la Mimas V2 debido a la configuración puesta en el FPGA. Sin embargo, al igual que en los casos anteriores se probará una representación alternativa para el sistema de manera de muestrear otro caso y ver si este circuito de verdad funciona. En este caso se planteará al bit cuántico menos significativo como el bit de control por lo que se pondrá a la variable C igual a 1. También para cambiar el contexto se va a hacer que tanto SBetaA y SAlfaB sean igual a 1. De manera de que se obtienen los siguientes resultados físicos del FPGA.

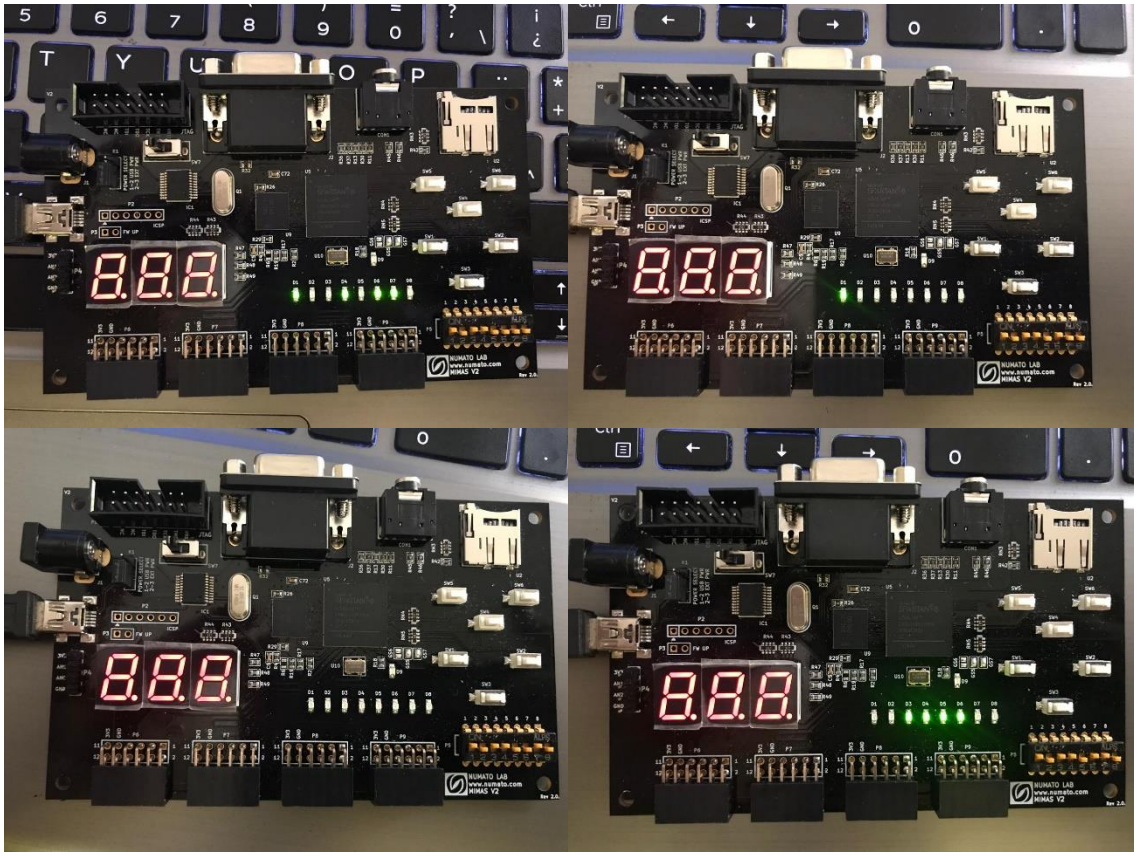


Ilustración 69. FPGA configurado con la compuerta CNOT: caso 2.

Fuente de la imagen: Elaboración propia.

Debido a los parámetros configurados para este caso se puede entonces hacer la siguiente representación matemática.

$$|yx\rangle = -\frac{1}{3}|00\rangle + \frac{2\sqrt{2}}{3}|01\rangle + 0|10\rangle - 0|11\rangle \rightarrow CNOT|yx\rangle = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} -\frac{1}{3} \\ \frac{2\sqrt{2}}{3} \\ 0 \\ -0 \end{pmatrix} = \begin{pmatrix} -\frac{1}{3} \\ -0 \\ 0 \\ \frac{2\sqrt{2}}{3} \end{pmatrix}$$

$$\approx \begin{pmatrix} -0.333 \\ -0 \\ 0 \\ 0.942 \end{pmatrix} = -0.333|00\rangle - 0|01\rangle + 0|10\rangle + 0.942|11\rangle$$

Teniendo estos valores se mapearán los sets de bits necesarios para la representación del escalar en base a la tabla 9. De tal forma entonces que el set de bits se ve de la siguiente manera.

$$\alpha = 0.333 \rightarrow 1001011, \qquad \beta = 0 \rightarrow 1000000, \qquad \lambda = 0 \rightarrow 0000000,$$
$$\delta = 0.942 \rightarrow 0011110$$

Obtenidos estos valores se puede observar que el circuito puesto en la FPGA reacciona de tal manera que debería, esto en comparación a la base matemática que se dedujo con la teoría de sustento descrita en el capítulo 3.

Ahora se planteará siempre la compuerta CNOT, pero aplicado a los tres bits cuánticos. De manera entonces que se va a configurar el FPGA con el circuito diseñado para esta aplicación de esta compuerta.

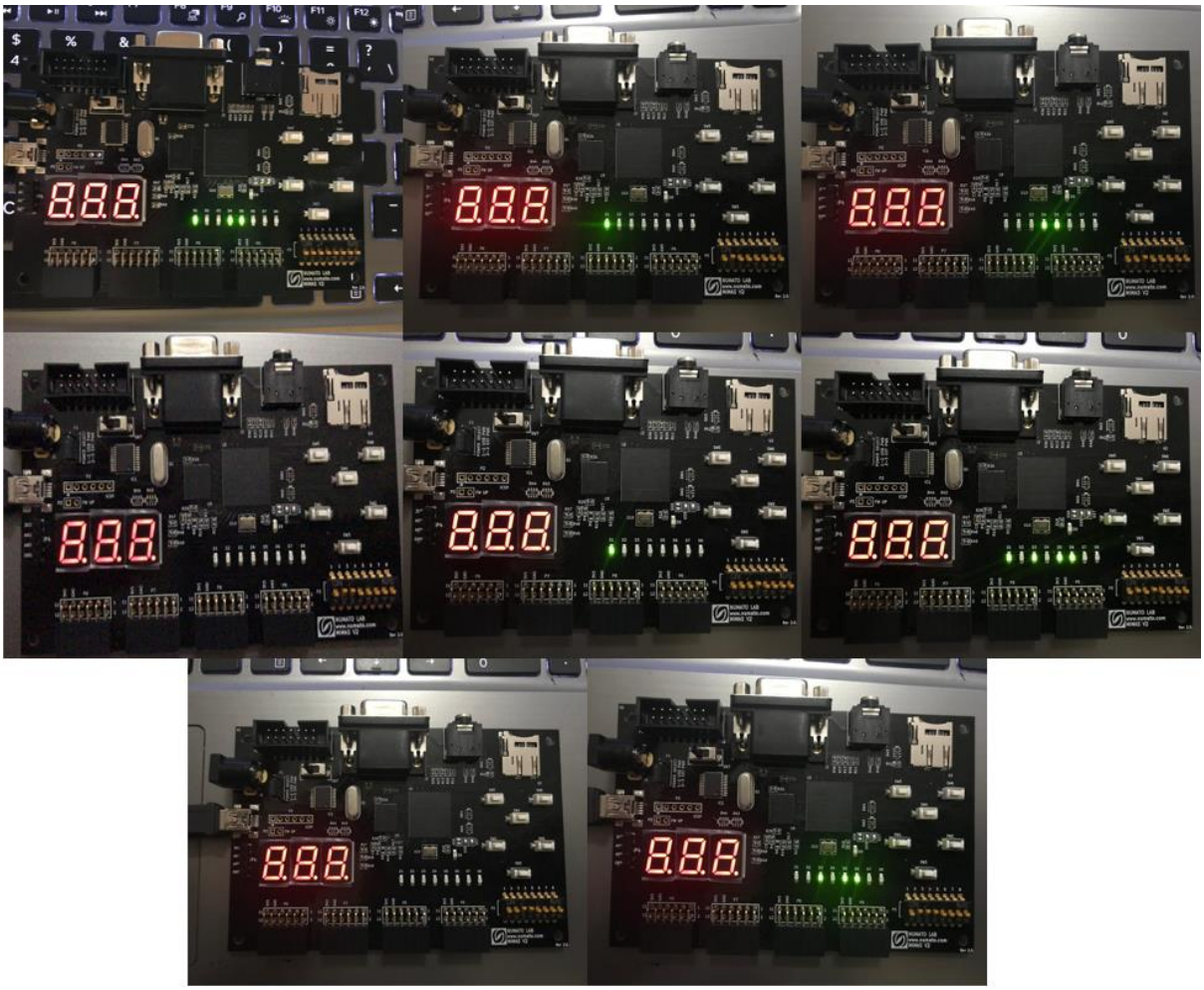


Ilustración 70. FPGA configurado con la compuerta CNOT aplicado a tres bits cuánticos: caso 1.

Fuente de la imagen: Elaboración propia.

En este caso se tiene el sistema de tres bits cuánticos planteado para la verificación de la funcionalidad de esta compuerta bajo esta aplicación. Si se recapitula los tres bits cuánticos bajo el orden de significancia se forma el siguiente sistema de tres bits cuánticos.

$$|xzy\rangle = -\frac{1}{3\sqrt{2}}|000\rangle - 0|001\rangle + \frac{1}{3\sqrt{2}}|010\rangle + 0|011\rangle + \frac{2}{3}|100\rangle + 0|101\rangle - \frac{2}{3}|110\rangle - 0|111\rangle$$

Si se pasa este sistema por la compuerta CNOT con la aplicación explicada para tres bits cuánticos, se obtiene la siguiente referencia matemática.

$$CNOT|xzy\rangle = -\frac{1}{3\sqrt{2}}|000\rangle - 0|001\rangle + \frac{1}{3\sqrt{2}}|010\rangle + 0|011\rangle - 0|100\rangle - \frac{2}{3}|101\rangle + 0|110\rangle + \frac{2}{3}|111\rangle$$

De manera entonces que los escalares de respuesta se deben de mapear de la siguiente manera.

$$\alpha = 0.236 \rightarrow 1001100, \quad \beta = 0 \rightarrow 1000000, \quad \gamma = 0.236 \rightarrow 0001100,$$
$$\delta = 0 \rightarrow 0000000, \quad \varepsilon = 0 \rightarrow 1000000, \quad \zeta = 0.666 \rightarrow 1010110,$$
$$\eta = 0 \rightarrow 000000, \quad \theta = 0.666 \rightarrow 0010110$$

Asi entonces pudiéndose observar que la respuesta es igual a la compuerta por lo que se puede deducir que el circuito diseñado para la simulación de la compuerta CNOT aplicada a tres bits cuánticos funciona como debe y además el FPGA es capaz de procesar, y con facilidad, esta configuración.

Ahora finalmente se estará comprobando la funcionalidad de la compuerta ContraCNOT, el circuito diseñado para dos bits cuánticos y para tres bits cuánticos. Por lo que se configurará el FPGA con el archivo BIN generado para la compuerta cuántica.

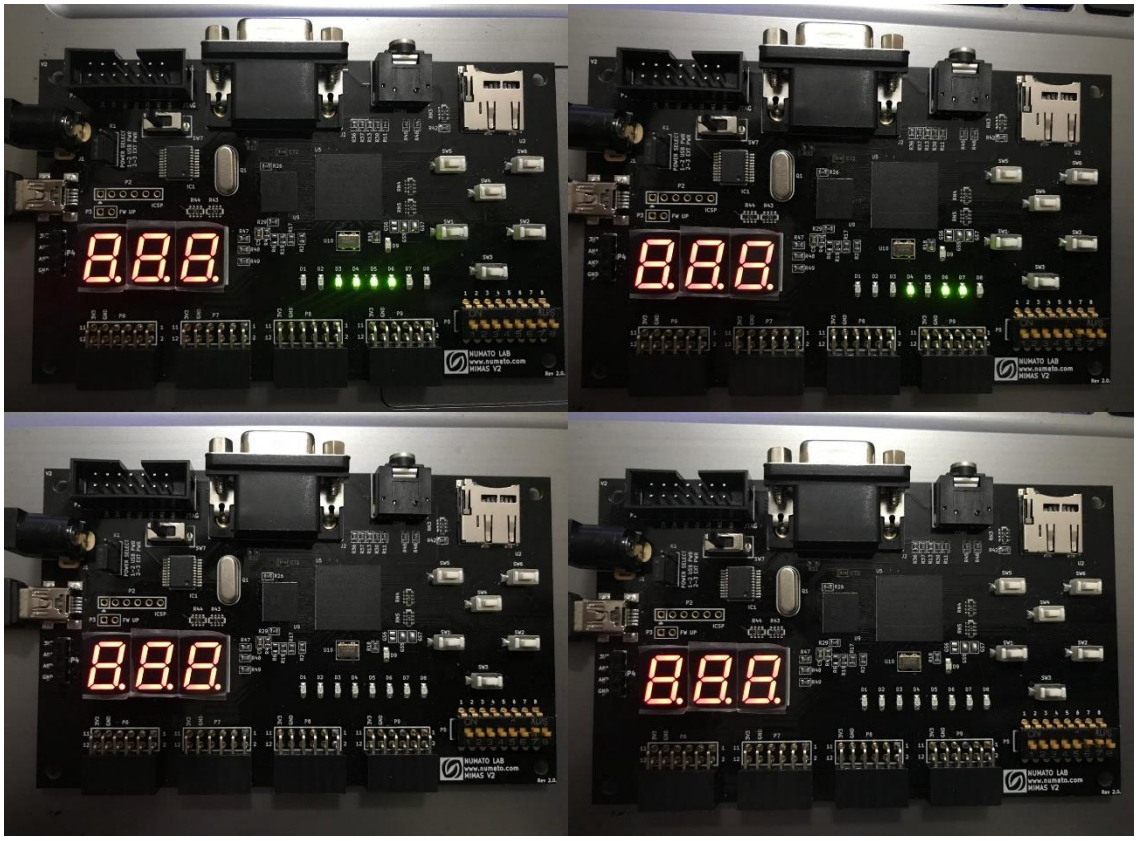


Ilustración 71. FPGA configurado con la compuerta ContraCNOT: caso 1.

Fuente de la imagen: Elaboración propia.

Considerando entonces este caso donde sí se ve los bits asignados para los signos de los escalares de entrada. Para lo que entonces se tiene el siguiente planteamiento matemático para este caso. También se recapitula el hecho de que el sistema de bits cuánticos de entrada es el mismo.

$$ContraCNOT|yx\rangle = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \frac{1}{3} \\ \frac{2\sqrt{2}}{3} \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} \frac{2\sqrt{2}}{3} \\ \frac{1}{3} \\ 0 \\ 0 \end{pmatrix} \approx \begin{pmatrix} 0.942 \\ 0.333 \\ 0 \\ 0 \end{pmatrix} = 0.942|00\rangle + 0.333|01\rangle$$

Teniendo estos valores se mapearán los sets de bits necesarios para la representación del escalar en base a la tabla 9. De tal forma entonces que el set de bits se ve de la siguiente manera.

$$\alpha = 0.942 \rightarrow 00011110, \qquad \beta = 0.333 \rightarrow 00001011, \qquad \lambda = 0 \rightarrow 0000000,$$
$$\delta = 0 \rightarrow 0000000$$

Estos sets de bits de respuesta se pueden ver que son los mismos que devuelven la FPGA por lo que entonces se puede ver que la compuerta cuántica está funcionando como debe. Sin embargo, se quiere al igual que en las otras compuertas, dar una prueba alternativa para la compuerta de manera de poder confirmar su buen funcionamiento. Por lo que se aplicara el siguiente caso. En el que el bit de control es el bit menos significativo y se hará que SAlfaA y SBetaB sean 1 por lo que estos escalares serán negativos.

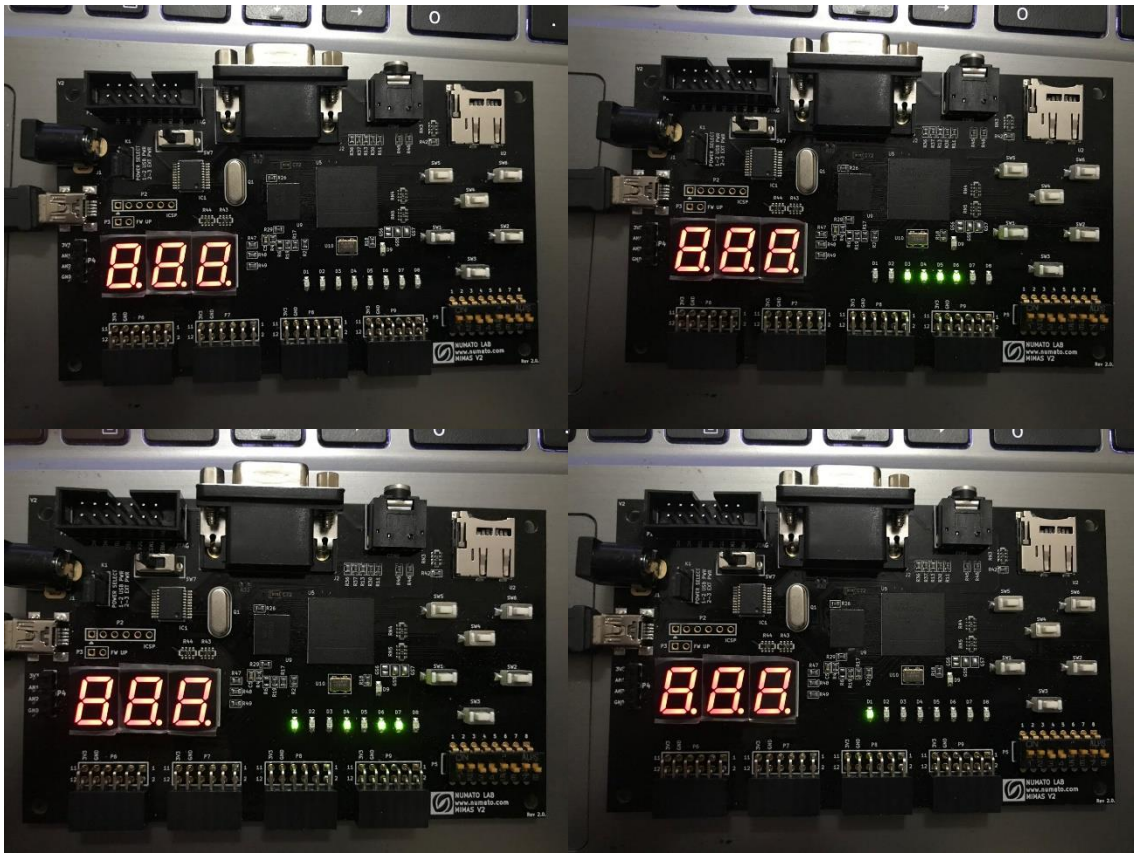


Ilustración 72. FPGA configurado con la compuerta ContraCNOT: caso 2.

Fuente de la imagen: Elaboración propia.

Planteando entonces este caso se puede llegar a la siguiente representación matemática.

$$|yx\rangle = -\frac{1}{3}|00\rangle + \frac{2\sqrt{2}}{3}|01\rangle + 0|10\rangle - 0|11\rangle \rightarrow ContraCNOT|yx\rangle = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} -\frac{1}{3} \\ \frac{2\sqrt{2}}{3} \\ 0 \\ -0 \end{pmatrix}$$

$$= \begin{pmatrix} 0 \\ \frac{2\sqrt{2}}{3} \\ -\frac{1}{3} \\ -0 \end{pmatrix} \approx \begin{pmatrix} 0 \\ 0.942 \\ -0.333 \\ -0 \end{pmatrix} = 0|00\rangle + 0.942|01\rangle - 0.333|10\rangle - 0|11\rangle$$

Teniendo estos valores se mapearán los sets de bits necesarios para la representación del escalar en base a la tabla 9. De tal forma entonces que el set de bits se ve de la siguiente manera.

$$\alpha = 0 \rightarrow 0000000, \qquad \beta = 0.942 \rightarrow 0011110, \qquad \lambda = 0.333 \rightarrow 1001011,$$
$$\delta = 0 \rightarrow 1000000$$

Obtenidos estos valores se puede observar que el circuito puesto en la FPGA reacciona de tal manera que debería, esto en comparación a la base matemática que se dedujo con la teoría de sustento descrita en el capítulo 3.

5.3 Segundo Ciclo

En este ciclo se estará emulando el algoritmo de Deutsch aplicado al problema de Deutsch. Se estará analizando el circuito de manera de plantear las herramientas necesarias para la emulación, saber los puntos clave para la verificación del funcionamiento de la emulación, y saber los parámetros a seguir para el diseño del circuito equivalente para la representación del algoritmo.

5.3.1 Análisis del Algoritmo de Deutsch aplicado al Problema de Deutsch

El problema de Deutsch mapea un bit cuántico a un bit cuántico. Donde el objetivo es encontrar el tipo de función que se encuentra en lo que se denota un oráculo. Para eso se tiene que saber qué valor devuelve al final de algoritmo el segundo bit cuántico debido a un estado de entrada $|01\rangle$. Ahora cada uno de los cuatro posibles casos para el oráculo, verse la ecuación 51, cada caso tiene asignado una compuerta cuántica.

Cuando $f(x) = 0$ el circuito no pasa por ninguna compuerta. Cuando $f(x) = 1$ el primer bit cuántico en el circuito pasa por una compuerta NOT y el segundo bit cuántico no pasa por nada. Ahora en estos dos primeros casos planteados cada bit está funcionando por su cuenta en la equivalencia cuántica, ahora se debe fusionar, ya que matemáticamente estos dos bits cuánticos cuando ambos pasan por una compuerta cuántica estos realizan un producto tensor entre ellos. Para causar esto efecto en los circuitos hay que utilizar el circuito para fusionar dos bits cuánticos que se crea en la sección 5.1.2. En las siguientes dos compuertas de este circuito la fusión de bits cuánticos ya está implícito al principio del funcionamiento de la compuerta. Cuando $f(x) = x$ la compuerta es la compuerta CNOT, donde el bit de control es el más significativo. Y cuando $f(x) = \bar{x}$ la compuerta asignada la compuerta ContraCNOT, donde el bit de control igual es el más significativo. De manera entonces que se necesita tener un set de entradas en el que se pueda manipular que compuerta está metida en el circuito. Como ya se ha utilizado antes, se ha visto que el multiplexor logra manipular la información que pasa por los cables. Debido a que la entrada al multiplexor es de 4 distintas variables el multiplexor se tiene que manipular con dos bits, de manera de utilizar las cuatro posibles combinaciones para la manipulación de la variable en uso. También si se recapitula el algoritmo de Deutsch se puede observar que ambos bits cuánticos tienen que pasar por una compuerta de Haddamard antes de que estos pasen por el oráculo.

Explicado entonces la manera de controlar el oráculo y el paso en el algoritmo previo a esta función, se debe de ser capaces de analizar tan solo un bit cuántico de la respuesta que se obtendrá de la salida del multiplexor que controla la compuerta en uso. Para esto se debe de generar una compuerta capaz de separar los bits cuánticos.

Matemáticamente hablando es muy complicado saber el valor de los cuatro escalares de cada bit cuántico por separado a partir del valor de los valores de los escalares de los dos bits cuánticos ya expresados juntos. Esto debido a que se tiene un sistema de ecuaciones cuadráticas. Esto complica poder generalizar esta herramienta para cualquier caso posible. Ahora ya que se tiene la necesidad de un circuito de este estilo se tiene que buscar crearlo bajo ciertas circunstancias.

Si se toma entonces el caso que se tiene enfrente, se sabe que el valor que va a llegar a la función de fisión de bits cuánticos va a ser un valor entre 1 o 0, ya sea para el primer bit

cuántico en el que $\beta = 1$ y $\alpha = 0$ o en el caso contrario para el segundo bit cuántico. Se sabe que estos valores al pasar por la compuerta de Haddamard van a tener la misma magnitud, que es el valor $1/\sqrt{2}$ que mapeado va a ser el set de bits 0b010111, donde el bit que estaba en el estado |0⟩ va a devolver ambos escalares positivos, y el que estaba en estado |1⟩ va a devolver el escalar para el nuevo estado |0⟩ positivo y el escalar para el nuevo estado |1⟩ va a ser negativo. Sabiendo entonces la magnitud que va a tener previo a la fusión de los bits se puede deducir que el circuito al unirlo y multiplicar cada escalar debidamente va a devolver el mismo escalar para todos los cuatro estados del sistema de dos bits cuánticos, donde la magnitud de estos escalares va a ser $1/2$. Sabiendo entonces que la magnitud de todos los escalares previo al circuito para la fisión de bits cuánticos es $1/2$, se tiene que hacer que hacer que este circuito pueda agarrar estas cuatro entradas de $1/2$ del sistema de dos bits cuánticos y dividirlas en cuatro salidas, dos para cada bit cuántico ya por separado, de $1/\sqrt{2}$. Sin embargo, este circuito solo va a funcionar con la condición que sus entradas sean un sistema de dos bits cuánticos sean provenientes de una compuerta de Haddamard, donde si hubo otra compuerta de por medio debe ser una que no afecte la magnitud de los escalares solo la posición de estos, y previo a la compuerta de Haddamard ambos bits cuánticos o tuvieron que estar en estado |0⟩ o |1⟩.

Ahora la parte más importante al momento de dividir los bits cuánticos es saber que signo tiene cada escalar de los dos bits cuánticos siendo fisionados. Ya que estos definirán si el estado entrando es la base |+⟩ o la base |−⟩, verse las ecuaciones 28 y 29 para recordar los conceptos de estas bases. Estas bases |+⟩ y |−⟩ al volver a pasar por una compuerta de Haddamard, que es el siguiente paso en el algoritmo, vuelven a estar en estado |0⟩ o |1⟩, respectivamente. Entonces para ver el análisis de los signos se debe sintetizar las expresiones matemáticas para cada uno de los cuatro casos, para los que se tomará la expresión de la resolución del algoritmo a partir del punto antes de entrar al oráculo, tomando el oráculo como una de las compuertas en cada uno de los casos.

Caso 1:

En este caso se va a tomar el caso en el que $f(x) = 0$, donde el circuito es representado solo por el cable como tal, por asi decirlo.

$$\frac{1}{2}(|0\rangle + |1\rangle)(|0\rangle - |1\rangle) \rightarrow U_f \rightarrow \frac{1}{2}(-1)^{f(x)}(|0\rangle + |1\rangle)(|0\rangle - |1\rangle)$$
$$= \frac{1}{2}(-1)^0(|0\rangle + |1\rangle)(|0\rangle - |1\rangle) = \frac{1}{2}(|0\rangle + |1\rangle)(|0\rangle - |1\rangle)$$
$$= \left(\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle\right)\left(\frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle\right)$$

Como se puede ver entonces la expresión se puede observar que el signo para el α del bit cuántico más significativo es positivo, el signo para el β del bit cuántico más significativo es positivo, el signo para el α del bit cuántico menos significativo es positivo y el signo para el β del bit cuántico menos significativo es negativo. Pero lo que devuelve el multiplexor será los signos del estado bajo un sistema de dos bits cuánticos. De manera que la síntesis de la formula queda de la siguiente manera, donde no se va a tomar en cuenta las magnitudes solo los signos.

$$(|0\rangle + |1\rangle)(|0\rangle - |1\rangle) = |00\rangle - |01\rangle + |10\rangle - |11\rangle$$

Eso significa que se tiene que considerar el caso en el que los signos devuelven este set de bits, recordando que cada escalar tiene su bit de signo, que sería 0b0101. Este debería de mapear de manera que el bit más significativo tenga 0b00 como signos de sus escalares, donde el más significativo del set de bits recién presentados es el de α y el que queda es de β

Caso2:

En este caso se va a tomar el caso en el que $f(x) = 1$, donde el circuito es representado por el bit menos significativo pasando por una compuerta NOT y el más significativo es solamente el cable.

$$\frac{1}{2}(|0\rangle + |1\rangle)(|0\rangle - |1\rangle) \rightarrow U_f \rightarrow \frac{1}{2}(-1)^{f(x)}(|0\rangle + |1\rangle)(|0\rangle - |1\rangle)$$
$$= \frac{1}{2}(-1)^1(|0\rangle + |1\rangle)(|0\rangle - |1\rangle) = \frac{1}{2}(|0\rangle + |1\rangle)(-|0\rangle + |1\rangle)$$
$$= \left(\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle\right)\left(-\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle\right)$$

Si se ve la expresión se puede observar que el signo para el α del bit cuántico más significativo es positivo, el signo para el β del bit cuántico más significativo es positivo, el signo para el α del bit cuántico menos significativo es negativo y el signo para él β del bit

cuántico menos significativo es positivo. Sin embargo, igual que en el caso anterior el multiplexor va a devolver los signos para los estados del sistema de dos bits cuánticos. De manera que la síntesis de la formula queda de la siguiente manera, donde no se tomará en cuenta las magnitudes solo los signos.

$$(|0\rangle + |1\rangle)(-|0\rangle + |1\rangle) = -|00\rangle + |01\rangle - |10\rangle + |11\rangle$$

Por lo que se puede observar que para este caso que el multiplexor devuelva un set de bits para los signos de 0b1010 se debe mapear a un set de bits de 0b00 para el bit cuántico más significativo.

Caso 3:

En este caso se va a tomar el caso en el que $f(x) = x$, donde el circuito es representado por una compuerta CNOT donde el bit cuántico más significativo es el bit de control.

$$\frac{1}{2}(|0\rangle + |1\rangle)(|0\rangle - |1\rangle) \rightarrow U_f \rightarrow \frac{1}{2}(-1)^{f(x)}(|0\rangle + |1\rangle)(|0\rangle - |1\rangle)$$

$$= \frac{1}{2}\left((-1)^{f(x)}|0\rangle + (-1)^{f(x)}|1\rangle\right)(|0\rangle - |1\rangle)$$

$$= \frac{1}{2}((-1)^0|0\rangle + (-1)^1|1\rangle)(|0\rangle - |1\rangle) = \frac{1}{2}(|0\rangle - |1\rangle)(|0\rangle - |1\rangle)$$

$$= \left(\frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle\right)\left(\frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle\right)$$

Si se ve la expresión se puede observar que el signo para el α del bit cuántico más significativo es positivo, el signo para el β del bit cuántico más significativo es negativo, el signo para el α del bit cuántico menos significativo es positivo y el signo para el β del bit cuántico menos significativo es negativo. Sin embargo, igual que en el caso anterior el multiplexor va a devolver los signos para los estados del sistema de dos bits cuánticos. De manera que la síntesis de la formula queda de la siguiente manera, donde no se va a tomar en cuenta las magnitudes solo los signos.

$$(|0\rangle - |1\rangle)(|0\rangle - |1\rangle) = |00\rangle - |01\rangle - |10\rangle + |11\rangle$$

Por lo que se puede observar que para este caso en el que el multiplexor devuelve un set de bits para los signos de 0b0110 se debe mapear a un set de bits de 0b01 para el bit cuántico más significativo.

Caso 4:

En este caso se va a tomar el caso en el que $f(x) = \bar{x}$, donde el circuito es representado por una compuerta ContraCNOT donde el bit cuántico más significativo es el bit de control.

$$\frac{1}{2}(|0\rangle + |1\rangle)(|0\rangle - |1\rangle) \rightarrow U_f \rightarrow \frac{1}{2}(-1)^{f(x)}(|0\rangle + |1\rangle)(|0\rangle - |1\rangle)$$

$$= \frac{1}{2}\left((-1)^{f(x)}|0\rangle + (-1)^{f(x)}|1\rangle\right)(|0\rangle - |1\rangle)$$

$$= \frac{1}{2}((-1)^1|0\rangle + (-1)^0|1\rangle)(|0\rangle - |1\rangle) = \frac{1}{2}(-|0\rangle + |1\rangle)(|0\rangle - |1\rangle)$$

$$= \frac{1}{2}(|0\rangle - |1\rangle)(-|0\rangle + |1\rangle) = \left(\frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle\right)\left(-\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle\right)$$

Como se puede ver la expresión se puede observar que el signo para el α del bit cuántico más significativo es positivo, el signo para el β del bit cuántico más significativo es negativo, el signo para el α del bit cuántico menos significativo es positivo y el signo para el β del bit cuántico menos significativo es negativo. Sin embargo, igual que en el caso anterior el multiplexor va a devolver los signos para los estados del sistema de dos bits cuánticos. De manera que la síntesis de la formula queda de la siguiente manera, donde no se va a tomar en cuenta las magnitudes solo los signos.

$$(|0\rangle - |1\rangle)(-|0\rangle + |1\rangle) = -|00\rangle + |01\rangle + |10\rangle - |11\rangle$$

Por lo que se puede observar que para este caso que el multiplexor devuelva un set de bits para los signos de 0b1001 se debe mapear a un set de bits de 0b01 para el bit cuántico más significativo.

Planteado todos los casos se puede observar que si la función es el caso 1 o 2 los signos que devolverá son positivos para los dos escalares del bit cuántico. Si la función es el caso 3 0 4 los signos que devolverá son positivos para el escalar del estado $|0\rangle$ y negativo para el escalar del estado $|1\rangle$. Por lo que se va a generar un circuito lógico que devuelva 1 cuando la entrada sea el set de bits que devuelve debido a los casos 3 y 4. De manera que la expresión booleana es la siguiente, valor de salida se representará con una e.

$$e = (SAlfa * SDelta) + (SBeta * SLamda)$$

Ecuación 74. Expresión booleana para separador de signos de dos bits cuánticos.

Entonces se pondrá en un multiplexor dos casos distintos que se estará manejando con la variable e, cuando esta sea 0 se quiere dejar pasar el set de bits 0b00 y cuando este sea 1 se quiere dejar el set de bits 0b01. De esta manera entonces tener los signos de cada uno de los escalares del bit más significativo del algoritmo pos función U_f.

Por último, previo a la medición, se pasará el bit más significativo por una compuerta de Haddamard, utilizando la magnitud obtenida del circuito para fisionar bits cuánticos y utilizando los signos obtenidos por el circuito previamente explicado. Luego de esto se va a comparar los valores de los escalares α y β, del bit cuántico pasado por la compuerta de Haddamard. Si $\alpha > \beta$ es porque el estado que se tiene es $|0\rangle$, y si $\alpha < \beta$ es porque el estado que se tiene es $|1\rangle$. Por lo que entonces se va a estar midiendo estos valores por medio de unas luces LEDs. Dicho esto, entonces se necesita distribuir las señales de manera que los dos LEDs que se estarán utilizando se enciendan dependiendo del estado. Se va a controlar este paso de información con un multiplexor el cual tendrá como controlador el comparador entre α y β, comparador que devolverá 1 si $\alpha < \beta$, entonces cuando este comparador sea 0 se busca que el multiplexor deje pasar los bits del estado $|0\rangle$ y si el comparador devuelve un 1 que el multiplexor devuelva los bits del estado $|1\rangle$.

5.3.2 Diseño de los circuitos para la emulación del Algoritmo de Deutsch adaptado al Problema de Deutsch

A priori a la descripción del diseño del circuito para la emulación del algoritmo cuántico de Deutsch adaptado al problema de Deutsch se estará realizando el diseño del circuito separador de bits cuánticos. Este circuito busca separar los bits cuánticos. Ahora el diseño debe hacer que a partir de las entradas de los escalares, de $1/2$, conseguir que las salidas sean $1/\sqrt{2}$. Para saber cómo plantear esto en el diseño se debe primero hacer las matemáticas necesarias para saber qué proceso se puede realizar para este valor cambie.

$$\frac{1}{2} = \frac{1}{\sqrt{2}} * \frac{1}{\sqrt{2}}$$

Por lo que si se despeja para $1/\sqrt{2}$, se obtiene la siguiente expresión.

$$\frac{1}{\sqrt{2}} = \frac{1}{2} * \sqrt{2}$$

Sin embargo, la raíz cuadrada de dos es un valor mayor a 1 y el multiplicador no acepta valores de este tipo por lo que entonces se tendrá que manejar un valor mayor. Por lo que se puede realizar lo siguiente.

$$\frac{1}{\sqrt{2}} = \frac{1}{2} * \frac{\sqrt{2}}{2} + \frac{1}{2} * \frac{\sqrt{2}}{2}$$

Esta expresión permite tomar el $1/2$ de entrada y multiplicarlo por un número menor a 1 y luego se podrá tomar ese valor después de la multiplicación y sumarlo a si mismo de manera de conseguir la salida buscada. Sabiendo esto entonces el circuito para la fisión de dos bits cuánticos, bajo las condiciones planteadas, se ve de la siguiente forma.

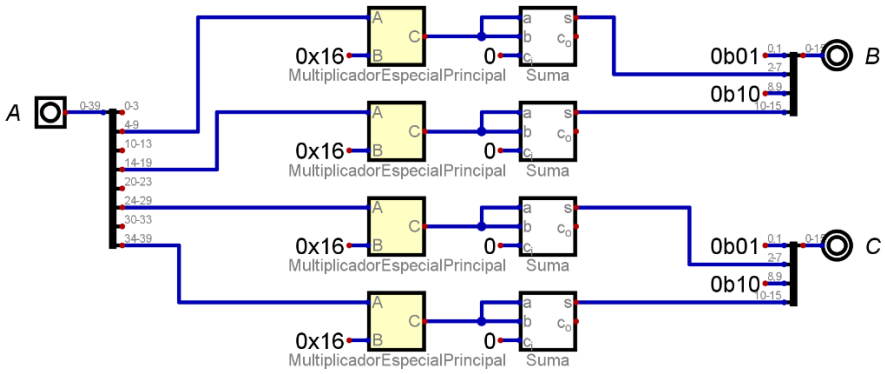


Ilustración 73. Circuito separador de dos bits cuánticos.

Fuente de la imagen: Elaboración propia.

Finalmente, para la separación de bits cuánticos, se va a aplicar la expresión booleana planteada en la ecuación 77. De manera que el circuito adaptado a esta expresión es el siguiente.

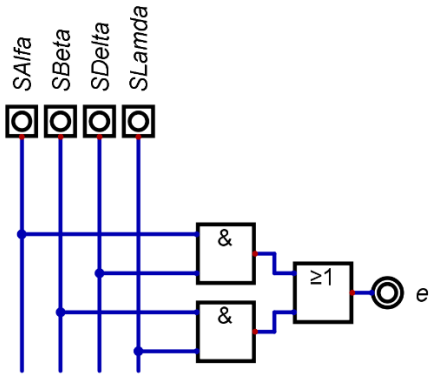


Ilustración 74. Circuito equivalente para la separación de signos de dos bits cuánticos.

Fuente de la imagen: Elaboración propia.

Teniendo entonces todas las herramientas necesarias, como ser compuertas cuánticas, separadores, etc., se puede ya plantear el diseño del circuito equivalente para el algoritmo de Deutsch aplicado al problema de Deutsch. El diseño del circuito equivalente es el siguiente.

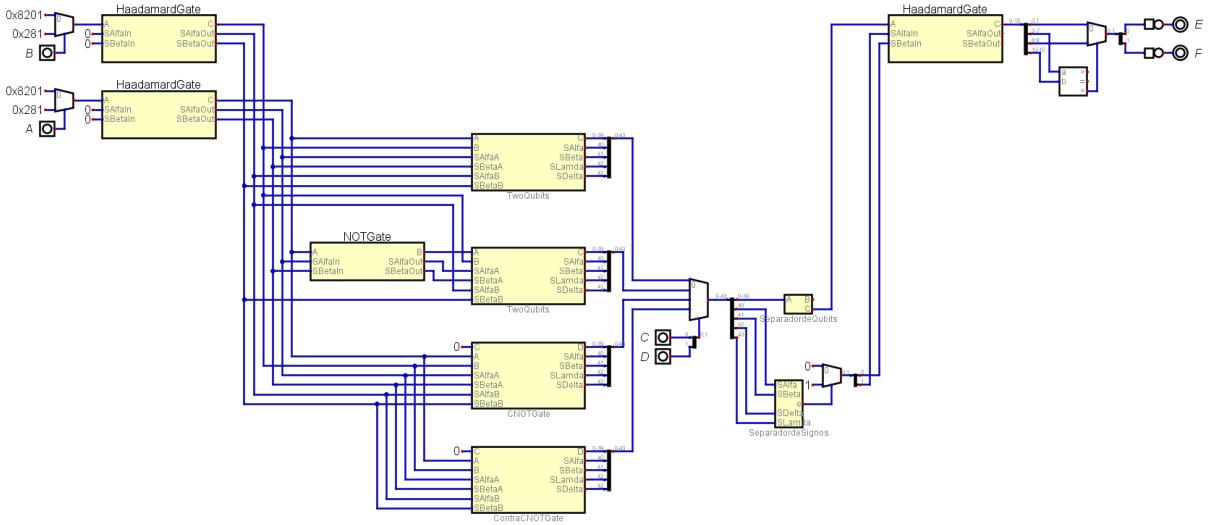


Ilustración 75. Circuito equivalente para la emulación del algoritmo de Deutsch adaptado al problema de Deutsch.

Fuente de la imagen: Elaboración propia.

El circuito fue diseñado de manera que los bits cuánticos pasaran por el mismo protocolo que el algoritmo planteado originalmente por Deutsch. De manera que lo se hace pasar por la compuerta de Haddamard ambos bits cuánticos, que ambos se estarán manejando con un pulso que son las entras A, para el bit cuántico menos significativo, y B, para el bit cuántico más significativo. Luego estos bits cuánticos pasan por el oráculo, de manera que en el circuito los bits cuánticos pasan por los circuitos necesarios para los cuatro posibles casos, el paso de información en el oráculo se va a estar manejando con dos entradas C y D. Donde sí C y D son 0b00 ocurra el primer caso, sí son 0b01 ocurra el segundo caso, sí son 0b10 ocurra el tercer caso y sí son 0b11 ocurra el cuarto caso. El siguiente paso es pasar la salida del multiplexor por un separador de bits cuánticos y de signos de manera de poder manipular cada bit por separado y poder tomar solo el bit más significativo. Luego solamente a este bit cuántico se pasará por una compuerta de

Haddamard. Finalmente se realiza la medición, que se hará con dos salidas, que en la Mimas V2 se estará representando como luces LEDs, en las que se controlan los pulsos de dos bits de manera que estos dos bits van a ser los del estado $|0\rangle$ en caso que el valor de α sea mayor que β y los dos bits van a ser los del estado $|1\rangle$ en el caso contrario, realizando esto con un comparador y un multiplexor.

5.3.2 Generar Códigos para los Diseños

Para generar los códigos en Verilog se van a repetir ciertos pasos realizados en el ciclo anterior. Como ya se sabe, el primer paso es exportar a Verilog desde el archivo del circuito equivalente del algoritmo cuántico de Deutsch en el software Digital. Para esto se usará la función del software de ese mismo nombre para generar el código necesario para poder correr en el software ISE Design Suite. Una vez entonces tenido el código exportado a Verilog y corrido ya en ISE Design Suite se procede a utilizar la función Implement Top Module, de manera de revisar que la implementación del diseño, la síntesis del circuito y la sintaxis del código estén correctas.

Una vez con el código compilado en su totalidad se procede a asignar los pines y generar el archivo UCF. Como se ve en la ilustración 55 se tiene 4 entradas y 2 salidas. Asi que se puede esperar que el archivo sea corto y sencillo. Las entradas, al igual que en las equivalencias de las compuertas cuánticas, se usará el dip switch disponible en la Mimas V2. Para las salidas se estarán usando dos luces LEDs también ya integradas en la placa de prototipaje. Sabiendo ya los pines también ya descritos para estos, en base al datasheet de la Mimas V2, el archivo UCF se ve de la siguiente manera.

```
1  NET "A" PULLUP;
2  NET "A" LOC = F17;
3  NET "B" PULLUP;
4  NET "B" LOC = F18;
5  NET "C" PULLUP;
6  NET "C" LOC = C18;
7  NET "D" PULLUP;
8  NET "D" LOC = C17;
9  NET "E" LOC = P15;
10 NET "F" LOC = P16;
```

Ilustración 76. Asignación de pines para las variables del algoritmo cuántico de Deutsch aplicado al problema de Deutsch.

Fuente de la imagen: Elaboración propia.

5.3.4 Pruebas de Funcionalidad

En esta sección se estará programando la FPGA y se observará si la FPGA es capaz de emular el circuito diseñado en las secciones anteriores para la emulación del algoritmo cuántico de Deutsch aplicado al problema de Deutsch. Luego se verá si este circuito se comporta como debe para simular un algoritmo cuántico. Primero entonces se subirá el archivo a la FPGA de la misma forma que se realizó para subir la programación de las compuertas.

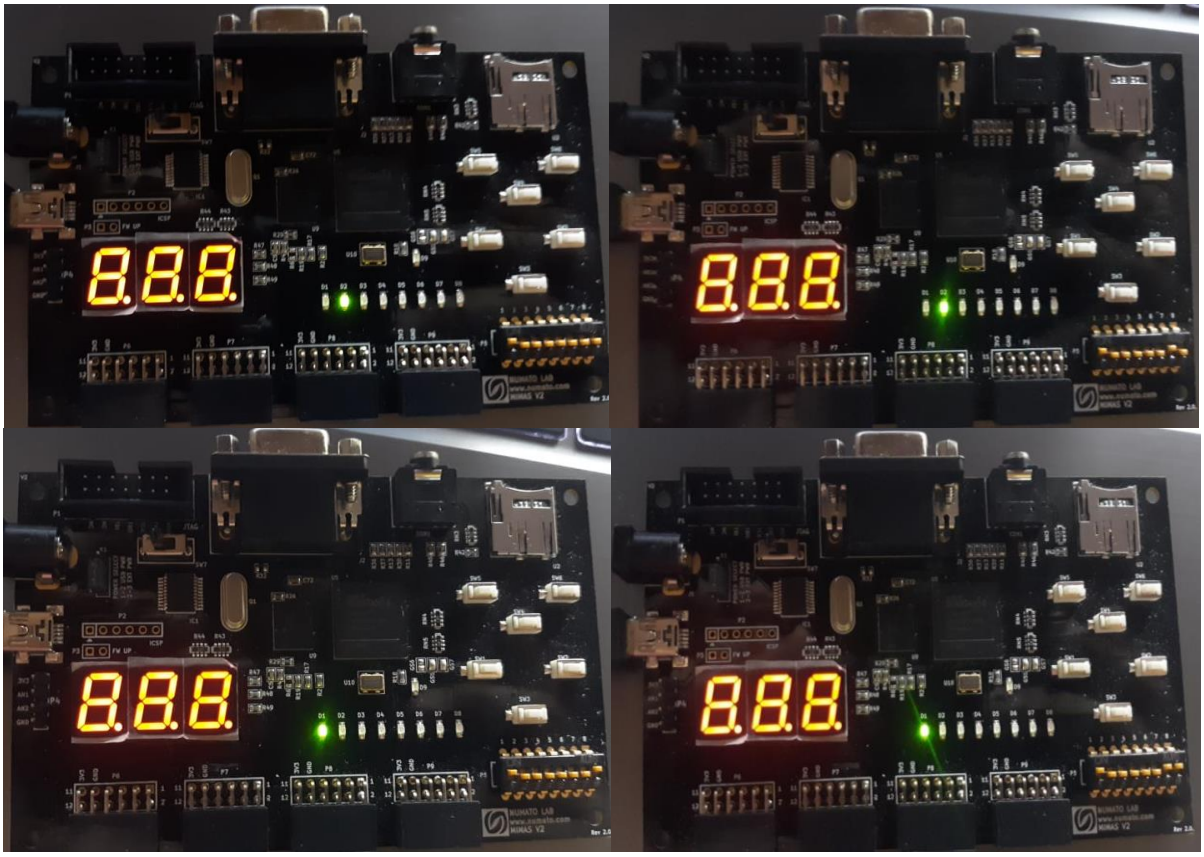


Ilustración 77. FPGA corriendo el circuito equivalente del algoritmo de Deutsch aplicado al problema de Deutsch.

Fuente de la imagen: Elaboración propia.

Primer se tiene que configurar la FPGA de manera que las entradas estén en el estado que deben según el algoritmo de Deutsch. En el cual se tiene al bit cuántico menos significativo siendo igual al estado $|1\rangle$ y el más significativo siendo igual al estado $|0\rangle$. Por lo que entonces se pone el interruptor 1, que según la declaración de variables este representa el bit cuántico menos significativo, encendido, y al interruptor 2, que representa al bit más significativo, apagado. Como se puede ver la FPGA devuelve el set de bits en base al estado

presente donde el LED D1 es el bit menos significativo del set de bit en representación del estado y el LED D2 es el bit más significativo del set de bit para la representación del estado. Como se puede ver cuando el dip switch está puesto para las funciones constantes, ósea cuando el interruptor 7 es 1 o 0 y el interruptor 8 es 0, este enciende el D2 y se mantiene apagado el D1, lo cual es la forma correcta que debe de presentar ya que cuando una función es constante el bit cuántico de respuesta está en estado |0⟩. Ahora en el caso en el que el dip switch está representando una función balanceada, ósea que el interruptor 7 es 1 o 0 y el interruptor 8 es 1, se enciende el D1 y se mantiene apagado el D2. Esto significa que se está devolviendo los valores correctos para la representación del estado |1⟩, estado que se mide en el bit de respuesta para las funciones balanceadas.

5.4 Tercer Ciclo

En este ciclo se estará emulando el algoritmo de Deutsch para la resolución del problema de Deutsch-Jozsa. Se analizará la aplicación del algoritmo bajo este problema de manera de ver si se cuenta con las herramientas necesarias. Una vez descritos los parámetros necesarios para la emulación se procederá a hacer el diseño en base a este análisis. Una vez teniendo los diseños se generará el código en Verilog para estos circuitos y un archivo UCF para la distribución de los pines en la placa Mimas V2. Finalmente, se verificará la funcionalidad del circuito y se pondrá en comparación para medir el rendimiento que tiene el FPGA para la emulación de este algoritmo bajo este problema.

5.4.1 Análisis del Algoritmo Deutsch bajo el Problema de Deutsch-Jozsa

Para empezar, se va a definir la idea de que se estará trabajando con tres bits cuánticos. Recordando el análisis matemático descrito en el marco teórico de esta aplicación se va a mapear una n cantidad de bits a un tan solo bit para verificar si la función en el oráculo es constante o balanceada. Esta n cantidad de bits cuánticos es uno menos que la cantidad de bits en total del sistema, ósea que la n será igual a dos. También hay que tener en cuenta que si los bits cuánticos de respuesta están todos en estado |0⟩ entonces la función es constante, en caso que al menos uno esté en estado |1⟩ la función ya no es constante y es balanceada.

Ahora para este problema la función es constante bajo los mismos parámetros que el problema anterior; $f(x) = 0$ cuando el circuito del oráculo no es nada más que los cables,

por asi decirlo; $f(x) = 1$ cuando en el circuito nada más el bit cuántico menos significativo pasa por una compuerta cuántica NOT y el resto de los bits cuánticos solo pasan por cable. Ahora para el caso en el que la función es balanceada se vuelve un tanto más complejo. En el problema anterior nada más se tiene un bit de entrada, aparte del bit adicional que se utiliza para el control de los signos. Debido a que hay un bit solo se tienen dos funciones posibles que es cuando el bit devuelve el valor que tiene el único bit que hay y el otro cuando el bit devuelve el valor contrario. En este caso en el problema de Deutsch-Jozsa con los parámetros de n siendo igual a dos se tienen más opciones de funciones donde se tendrá una mezcla de los bits cuánticos, siendo una suma directa entre los valores negados y no negados de ambos bits cuánticos.

Para esto entonces se va a estar trabajando con ocho casos distintos, los dos casos en los que función es constante y se van a plantear seis casos distintos en los que la función es balanceada. Previo a explicar los casos se plantea que el sistema de este algoritmo primero hace que los tres bits cuánticos pasen cada uno por una compuerta cuántica de Haddamard. Sabiendo esto entonces se pueden describir los casos a utilizar para el oráculo. Los seis casos cuando la función es balanceada son los siguientes.

1. El circuito cuántico pasa por un CNOT de dos bits que afecta al segundo bit más significativo y al bit cuántico menos significativo, donde el segundo bit cuántico más significativo es el bit de control y el menos significativo es el controlado.
2. Este caso se planteará igual al primero con la diferencia que el segundo bit cuántico más significativo previo a la entrada a la compuerta cuántica CNOT pasa por una compuerta cuántica NOT.
3. El siguiente caso será que los tres bits cuánticos pasen por una compuerta cuántica ContraCNOT donde el bit cuántico de control es el segundo más significativo y los controlados son los otros dos bits cuánticos.
4. Este caso será en el que los tres bits cuánticos pasen por una compuerta cuántica CNOT donde el bit de control es el más significativo y los otros dos bits cuánticos son los controlados.
5. Este caso es el mismo que el anterior, solo con una pequeña diferencia, y es que el bit cuántico de control es el segundo bit cuántico más significativo.

6. El último caso que s aplicará es el mismo caso que el 4 con la diferencia que el bit cuántico de control es el bit cuántico más significativo.

Estos casos se estarán controlando con un multiplexor, este multiplexor se estará controlando con tres pulsos distintos. Una vez entonces planteados los casos que se manejarán para la implementación de este problema de Deutsch-Jozsa al algoritmo de Deutsch se procede a hacer el análisis de los valores de los escalares y signos que devuelven los casos ya planteados. Por lo que también se quiere dividir el sistema en los bits cuánticos por individual.

Para hacer eso primero se verá cómo se puede dividir los valores de los escalares. Recordando que llegados a este punto todos los valores son iguales ya que todos provienen de una compuerta de Haddamard y previo a esto tenían un estado fijo. Por lo que la compuerta los transformo a una superposición con los mismos valores para cada escalar, pero lo importante es el análisis de los signos. Los escalares entonces en la representación matemática en el sistema ya unido de los tres bits cuánticos tiene un valor de $1/2\sqrt{2}$. Por lo que se busca mapear este valor que tienen los ocho escalares del sistema de tres bits cuánticos a repartir $1/\sqrt{2}$ a cada uno de los 6 estados que se van a tener de respuesta por los tres bits cuánticos que se quiere de respuesta. Para esto se puede observar que el valor $1/2\sqrt{2}$ es la mitad del valor $1/\sqrt{2}$ por lo que entonces nada más se sumarán el valor devuelto por los escalares unidos entre ellos y estos devolverán la aproximación del valor $1/\sqrt{2}$.

Ahora para los signos se tendrá que hacer un análisis un poco más profundo de manera de poder generalizar las salidas con respecto a las entradas. De manera que se sabe que uno de los bits siempre va a tener el mismo set de bits de signos para su superposición debido a la representación matemática y es el bit cuántico menos significativo este siempre tendrá los estados bajo el set de signos $|0\rangle - |1\rangle$. Debido a esto se planteará todos los casos posibles de signos para los otros dos bits cuánticos. Se realizará la multiplicación de los estados con sus signos, no se tomará en cuenta los escalares, y se tendría que ver el conjunto de signos que se obtienen al final de multiplicar los tres bits cuánticos. Luego viendo a este análisis se mapeará la entrada, que serán cualquiera de esas resoluciones de las multiplicaciones, a las salidas que serán 6 bits donde se va a distribuir cada bit a su respectivo bit cuántico de

respuesta y a su escalar en el bit cuántico que corresponda de tal manera entonces que se tienen los siguientes casos.

$$(|0\rangle + |1\rangle)(|0\rangle + |1\rangle)(|0\rangle - |1\rangle)$$
$$= (|000\rangle - |001\rangle + |010\rangle - |011\rangle + |100\rangle - |101\rangle + |110\rangle - |111\rangle)$$

$$(|0\rangle + |1\rangle)(|0\rangle - |1\rangle)(|0\rangle - |1\rangle)$$
$$= (|000\rangle - |001\rangle - |010\rangle + |011\rangle + |100\rangle - |101\rangle - |110\rangle + |111\rangle)$$

$$(|0\rangle + |1\rangle)(-|0\rangle + |1\rangle)(|0\rangle - |1\rangle)$$
$$= (-|000\rangle + |001\rangle + |010\rangle - |011\rangle - |100\rangle + |101\rangle + |110\rangle - |111\rangle)$$

$$(|0\rangle + |1\rangle)(-|0\rangle - |1\rangle)(|0\rangle - |1\rangle)$$
$$= (-|000\rangle + |001\rangle - |010\rangle + |011\rangle - |100\rangle + |101\rangle - |110\rangle + |111\rangle)$$

$$(|0\rangle - |1\rangle)(|0\rangle + |1\rangle)(|0\rangle - |1\rangle)$$
$$= (|000\rangle - |001\rangle + |010\rangle - |011\rangle - |100\rangle + |101\rangle - |110\rangle + |111\rangle)$$

$$(|0\rangle - |1\rangle)(|0\rangle - |1\rangle)(|0\rangle - |1\rangle)$$
$$= (|000\rangle - |001\rangle - |010\rangle + |011\rangle - |100\rangle + |101\rangle + |110\rangle - |111\rangle)$$

$$(|0\rangle - |1\rangle)(-|0\rangle + |1\rangle)(|0\rangle - |1\rangle)$$
$$= (-|000\rangle + |001\rangle + |010\rangle - |011\rangle + |100\rangle - |101\rangle - |110\rangle + |111\rangle)$$

$$(|0\rangle - |1\rangle)(-|0\rangle - |1\rangle)(|0\rangle - |1\rangle)$$
$$= (-|000\rangle + |001\rangle - |010\rangle + |011\rangle + |100\rangle - |101\rangle + |110\rangle - |111\rangle)$$

$$(|0\rangle + |1\rangle)(|0\rangle + |1\rangle)(|0\rangle - |1\rangle)$$
$$= (-|000\rangle + |001\rangle - |010\rangle + |011\rangle + |100\rangle - |101\rangle + |110\rangle - |111\rangle)$$

$$(|0\rangle + |1\rangle)(|0\rangle + |1\rangle)(|0\rangle - |1\rangle)$$
$$= (-|000\rangle + |001\rangle + |010\rangle - |011\rangle + |100\rangle - |101\rangle - |110\rangle + |111\rangle)$$

$$(|0\rangle + |1\rangle)(|0\rangle + |1\rangle)(|0\rangle - |1\rangle)$$
$$= (|000\rangle - |001\rangle - |010\rangle + |011\rangle - |100\rangle + |101\rangle + |110\rangle - |111\rangle)$$

$$(|0\rangle + |1\rangle)(|0\rangle + |1\rangle)(|0\rangle - |1\rangle)$$
$$= (|000\rangle - |001\rangle + |010\rangle - |011\rangle - |100\rangle + |101\rangle - |110\rangle + |111\rangle)$$

$$(|0\rangle + |1\rangle)(|0\rangle + |1\rangle)(|0\rangle - |1\rangle)$$
$$= (-|000\rangle + |001\rangle - |010\rangle + |011\rangle - |100\rangle + |101\rangle - |110\rangle + |111\rangle)$$

$$(|0\rangle + |1\rangle)(|0\rangle + |1\rangle)(|0\rangle - |1\rangle)$$
$$= (-|000\rangle + |001\rangle + |010\rangle - |011\rangle - |100\rangle + |101\rangle + |110\rangle - |111\rangle)$$

$$(|0\rangle + |1\rangle)(|0\rangle + |1\rangle)(|0\rangle - |1\rangle)$$
$$= (|000\rangle - |001\rangle - |010\rangle + |011\rangle + |100\rangle - |101\rangle - |110\rangle + |111\rangle)$$

$$(|0\rangle + |1\rangle)(|0\rangle + |1\rangle)(|0\rangle - |1\rangle)$$
$$= (|000\rangle - |001\rangle + |010\rangle - |011\rangle + |100\rangle - |101\rangle + |110\rangle - |111\rangle)$$

De manera entonces que basándonos en este análisis se planteará el set de bits posibles para la entrada y mapear este al set de bits de salida para los signos. El set de bits de entrada será de 8 bits donde el bit más significativo será el signo de α, y la salida será de 6 bits donde los bits serán seccionados de dos en dos donde el pares representan los signos de cada bit cuánticos respectivamente con su significancia. En el análisis anterior del juego con los signos se puede observar que cada caso para los estados del sistema de tres bits cuánticos se repite una vez. Por lo que entonces al momento de distribuir la respuesta se considerará el set de salida con respeto al que tenga menos signos negativos. Donde entonces el análisis del mapeo de los signos de salida con respecto a la entrada se ve de la siguiente manera.

$$0b01010101 \rightarrow 0b000001$$

$$0b01100110 \rightarrow 0b000101$$

$$0b10011001 \rightarrow 0b001001$$

$$0b10101010 \rightarrow 0b001101$$

$$0b01011010 \rightarrow 0b010001$$

$$0b01101001 \rightarrow 0b010101$$

$$0b10010110 \rightarrow 0b011001$$

$$0b10100101 \rightarrow 0b100001$$

Teniendo esto entonces se podrá dividir el sistema de tres bits cuánticos que se tendrá al salir del multiplexor a los tres bits cuánticos por separados. Teniendo ya entonces los tres bits cuánticos ya separados se procede a tomar los dos bits cuánticos más significativos y pasarlos por una compuerta de Haddamard. Finalmente, para la medición de los bits cuánticos se estará comparando los valores de los escalares de los estados de ambos bits

cuánticos. De manera entonces que si uno es mayor que el otro se encenderá un conjunto de dos LEDs dependiendo del estado que tenga un uno como escalar.

5.4.2 Diseño de los Circuitos Equivalentes para la Emulación del Algoritmo de Deutsch adaptado al problema de Deutsch-Jozsa

A priori al diseño del circuito equivalente para este ciclo se tienen que tener todas las herramientas diseñadas. Para este caso nada más hace falta el circuito necesario para la separación del sistema de tres bits cuánticos a los tres bits individuales. Aplicando entonces el análisis realizado en la sección anterior se puede realizar el siguiente circuito.

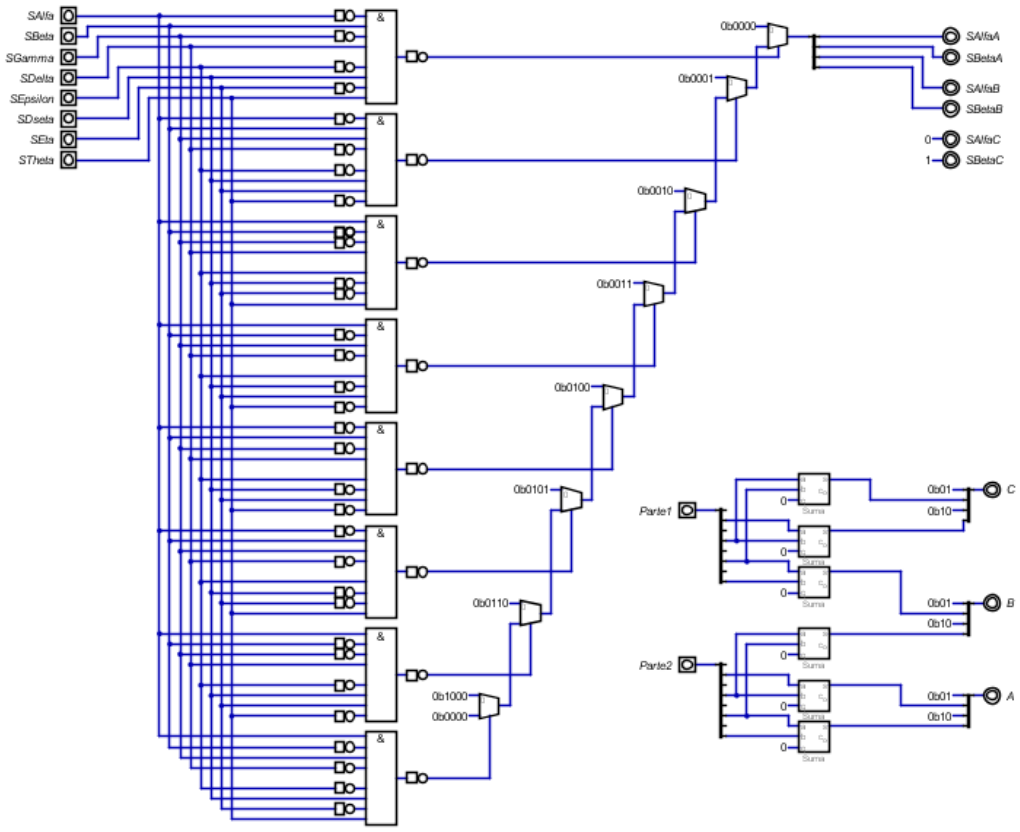


Ilustración 78. Separado de bits cuánticos para la representación de un sistema de tres bits cuánticos.

Fuente de la imagen: Elaboración propia.

Donde se considerará cada caso planteado para el juego de signos y debido a las entradas planteadas se planteará una compuerta clásica AND para estos valores. Cuando un valor sea 0 se puso un NOT clásico de manera de cumplir con el requisito del AND clásico. Planteando asi entonces los ocho casos la respuesta de estos ANDs clásicos se conectan a un

multiplexor que dejaba salir un set de 4 bits, esto debido a que como el set de bits para el bit cuántico menos significativo es el mismo siempre se descarta. Este set de 4 bits es dependiente de los sets de bits descritos en el análisis realizado en la sección anterior.

Teniendo entonces ya el separador de bits cuánticos para un sistema de tres bits cuánticos, se puede proseguir a plantear el diseño del algoritmo cuántico de Deutsch aplicado al problema de Deutsch-Jozsa, teniendo en cuanto, claro, los análisis previos. De tal forma que el circuito equivalente se ve de la siguiente forma.

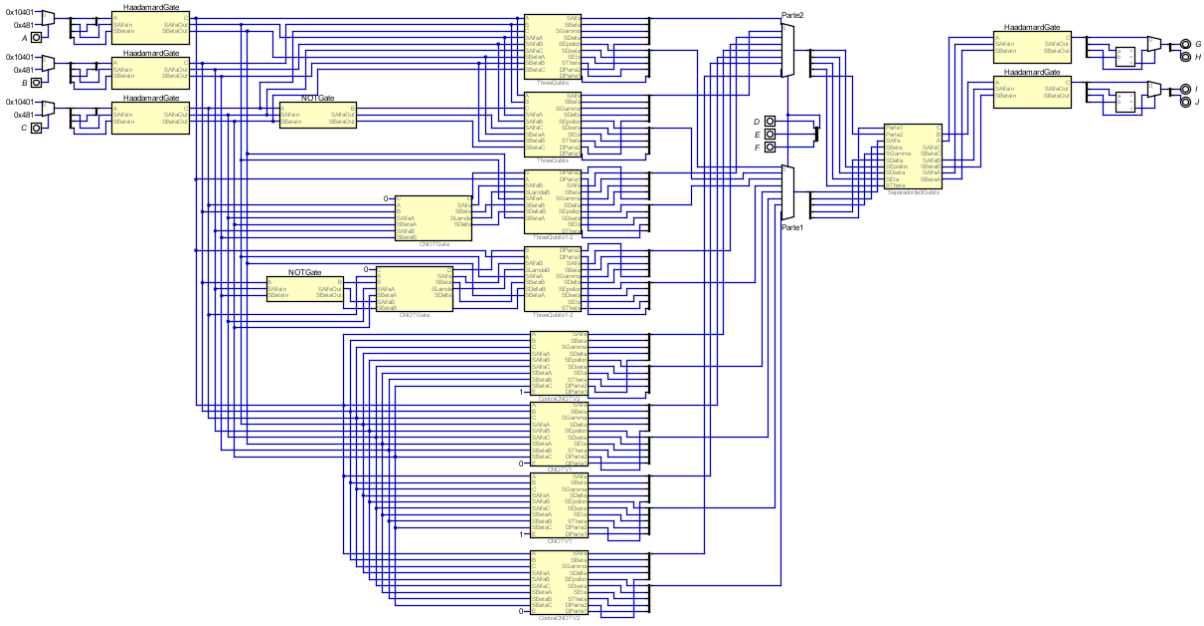


Ilustración 79. Diseño del circuito lógico equivalente del algoritmo de Deutsch aplicado al problema de Deutsch-Jozsa.

Fuente de la imagen: Elaboración propia.

El circuito debería de funcionar de la manera planteada en el análisis. Se tiene una entrada para cada uno de los tres bits cuánticos donde se controla el paso de un estado u otro con un simple pulso, esto realizado por medio de un multiplexor, recordando que el algoritmo pide que los primeros dos bits, o la n cantidad de bits, sean 0 y el bit menos significativo sea 0. Luego los bits cuánticos pasan por la representación de la compuerta de Haddamard. Al entrar al oráculo se va a poner cada uno de los datos a todos los casos planteados para la función del oráculo como tal. Controlando entonces con un multiplexor y tres bits para el selector de este multiplexor el paso de información dependiendo que caso se quiere considerar. Luego la salida del multiplexor se conecta al separador de bits

cuánticos para sistemas de tres bits cuánticos de manera de poder medir los dos bits cuánticos más significativos y poder tener la resolución de si la función es constante o balanceada.

5.4.3 Generar Códigos para los Diseños

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 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.

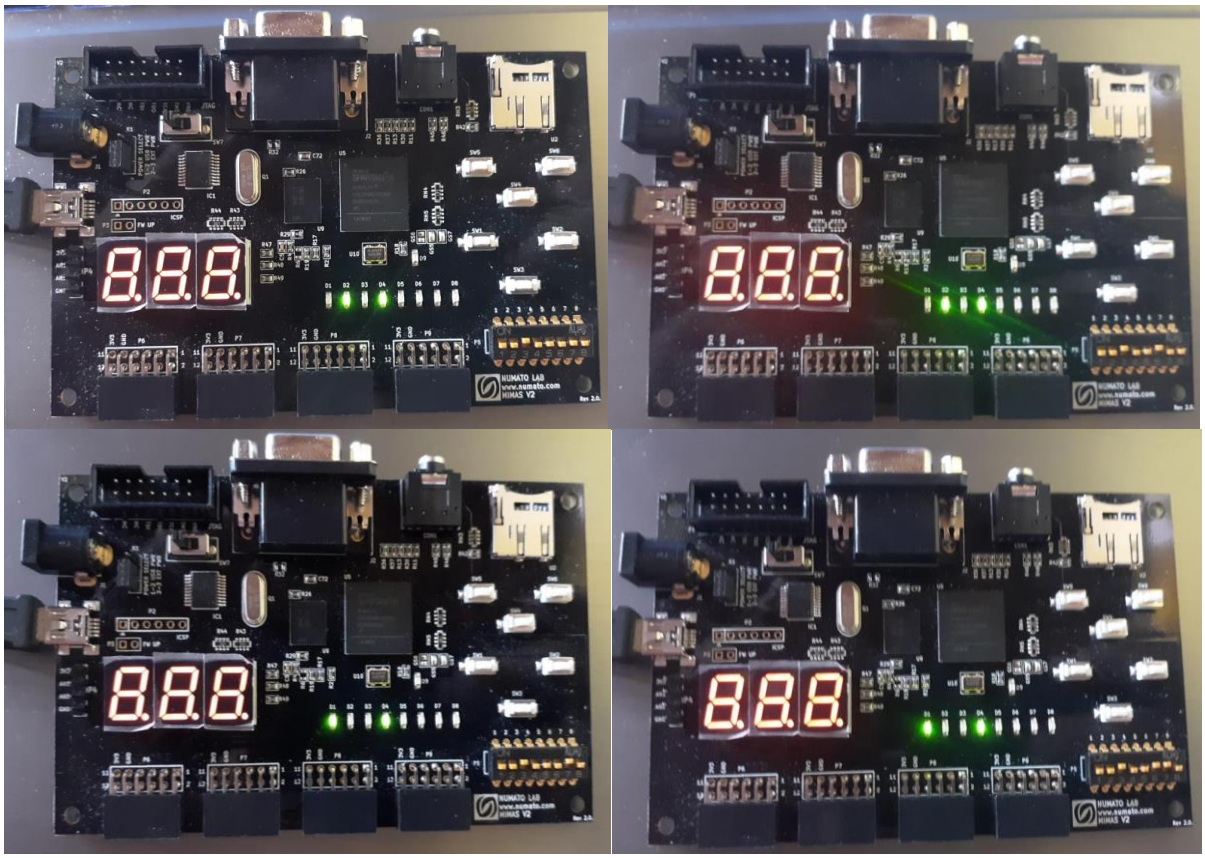
```
1  NET "A" PULLUP;
2  NET "A" LOC = F17;
3  NET "B" PULLUP;
4  NET "B" LOC = F18;
5  NET "C" PULLUP;
6  NET "C" LOC = E16;
7  NET "D" PULLUP;
8  NET "D" LOC = D17;
9  NET "E" PULLUP;
10 NET "E" LOC = C18;
11 NET "F" PULLUP;
12 NET "F" LOC = C17;
13 NET "G" LOC = P15;
14 NET "H" LOC = P16;
15 NET "I" LOC = N15;
16 NET "J" LOC = N16;
```

Ilustración 80. Asignación de pines para las variables del circuito equivalente del algoritmo de Deutsch aplicado al problema de Deutsch-Jozsa.

Fuente de la imagen: Elaboración propia.

5.4.4 Verificación de la Funcionalidad

En esta sección se estará verificando la funcionalidad de la FPGA al momento de emular el algoritmo cuántico de Deutsch aplicado al problema de Deutsch-Jozsa. Primero se va a generar el archivo BIN en el software ISE Design Suite. Generado el archivo BIN se sube a la FPGA por medio del software Mimas Configuration Tool. Subido entonces el archivo al FPGA se procede a medir si el FPGA es capaz de procesar el circuito diseñado para el algoritmo y luego ver si este trabaja de acuerdo a los parámetros necesarios para la emulación del algoritmo como tal.



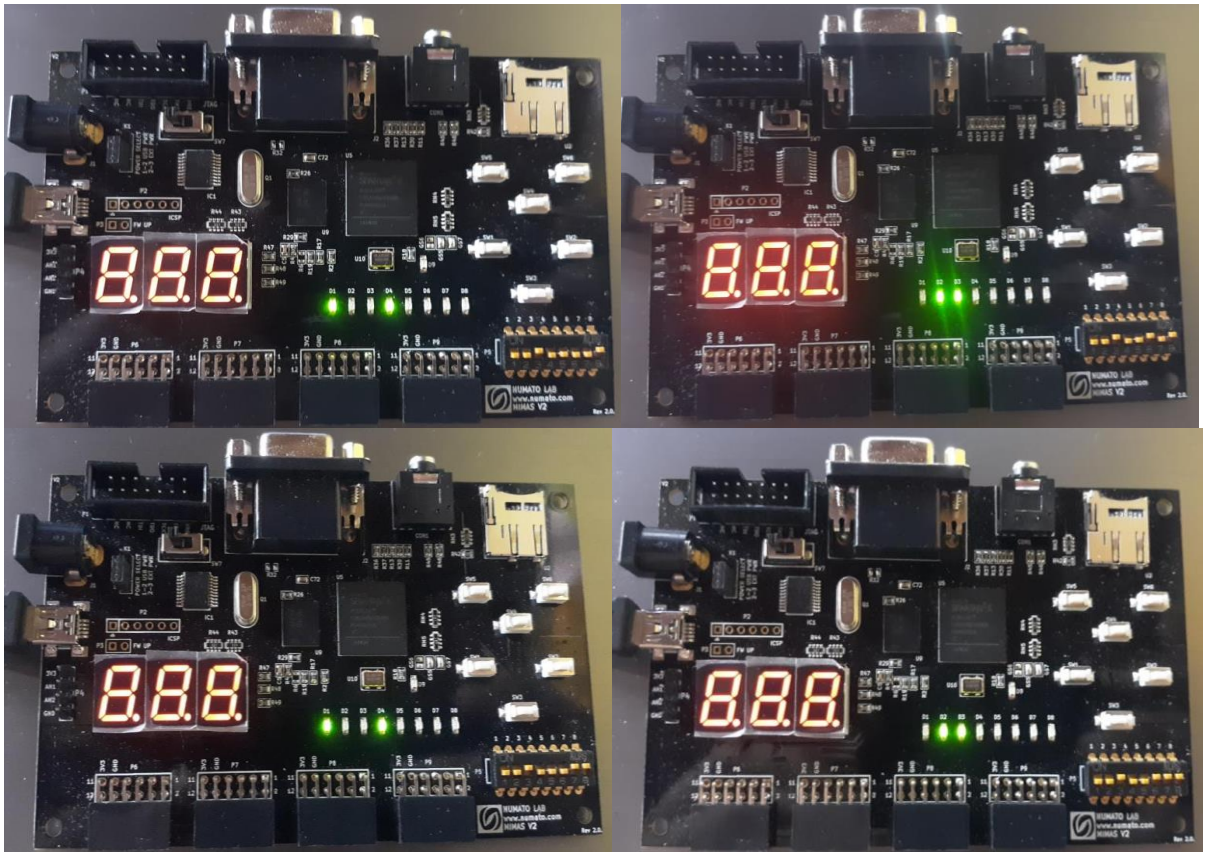


Ilustración 81. FPGA corriendo el circuito equivalente del algoritmo de Deutsch aplicado al problema de Deutsch.

Fuente de la imagen: Elaboración propia.

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 las imágenes se puede observar 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.

CAPÍTULO 6. CONCLUSIONES

1. Se logró implementar una FPGA para la emulación del algoritmo de Deutsch aplicado tanto al problema de Deutsch como al problema de Deutsch-Jozsa.
2. Se estableció una relación entre las compuertas cuánticas y las características cuánticas, concepto que se logró aplicar a los circuitos diseñados.
3. Se logró diseñar los circuitos lógicos digitales equivalentes para la simulación de las resoluciones de los problemas propuestos para la investigación.
4. Se estandarizaron los diseños realizados para los circuitos lógicos digitales equivalentes para cualquier escenario posible, esto bajo los parámetros establecidos para la representación de los bits cuánticos.

CAPÍTULO 7. RECOMENDACIONES

La presente tesis deja un gran campo abierto para futuros proyectos de investigación en cuanto simulación e implementación de la computación cuántica. Y sea para la implementación en un FPGA o solo para el simple diseño de los circuitos lógicos equivalentes para ciertos componentes cuánticos. Se puede entonces ampliar el proyecto de manera que se pueden aplicar más bits cuánticos para el algoritmo de Deutsch aplicado al problema de Deutsch-Jozsa. También los diseños establecidos pueden ayudar a la resolución a los problemas de Bernstein Vazirani y Simon. De igual forma quedo expuesto el algoritmo de Shor el cual tiene una importante aplicación, ya que se utiliza para decodificar el algoritmo de RSA. sin embargo, para este algoritmo es muy probable que se necesite una gamma más tanto de FPGA como de lenguaje de programación.

GLOSARIO

1. **ALGORITMO:** Cualquier procedimiento computacional bien definido que toma algún valor, o conjunto de valores, como entrada y produce otro valor, o conjunto de valores, como salida.
2. **ALGORITMO CUÁNTICO:** Es un algoritmo que funciona bajo los lineamientos y modelado de la computación cuántica.
3. **ALGORITMO DE DEUTSCH:** Es un algoritmo cuántico, nombrado tras su inventor David Deutsch, que se aplica para encontrar que tipo de función, ya sea balanceada o constante, devuelve un sistema.
4. **ALGORITMO DE RSA:** Es un algoritmo clásico, nombrado tras las iniciales de sus tres creadores, que se utiliza para la codificación, por medio una clave publica, y la decodificación, por medio de una llave privada, de un mensaje.
5. **ALGORITMO DE SHOR:** Es un algoritmo cuántico, nombrado por su creado Peter Shor, diseñado para factorizar números grandes en un tiempo polinomial, una de sus principales aplicaciones es la decodificación del algoritmo de RSA.
6. **BIT CUÁNTICO:** Es el sistema de medición de la información cuántica que maneja dos estados distintos y que se puede regir la probabilidad de que sea uno de estos dos estados arbitrariamente.
7. **CIRCUITO CUÁNTICO:** Es el esquema de la posición de los bits cuánticos y las compuertas cuánticas en el flujo de un algoritmo cuántico.
8. **CIRCUITOS INTEGRADOS:** Es un conjunto de transistores miniaturizados, normalmente puestos sobre una capa de silicón, que están conectados internamente diseñados para realizar funciones computacionales.
9. **COMPUERTA CUÁNTICA:** Es una entidad cuántica con la función de manipular los escalares de los estados de manera de conseguir una salida distinta a la entrada; matemáticamente estas son matrices unitarias.
10. **COMPUERTA CUÁNTICA CNOT:** Es una compuerta cuántica diseñada para manipular dos bits cuánticos, uno de control y uno controlado, al estar en estado uno el bit de control hace que el bit controlado cambie de estado.

11. **COMPUERTA CUÁNTICA CONTRACNOT:** Es una compuerta cuántica con la misma funcionalidad que la compuerta CNOT con la diferencia que el cambio se da cuando el bit de control está en estado cero.
12. **COMPUERTA CUÁNTICA DE HADDAMARD:** Es una compuerta cuántica diseñada para hacer rotar a un bit cuántico 45°; una de sus características mas notables es que al pasarle un bit cuántico en estado fijo esta lo pone en estado de superposición.
13. **COMPUERTA CUÁNTICA NOT:** Es una compuerta cuántica que busca cambiar de estado a un bit cuántico.
14. **COMPUTACIÓN CUÁNTICA:** Es un tipo de computación que utiliza los conceptos como ser el entrelazamiento y la superposición para poder realizar funciones computacionales, cuya información se rige por el bit cuántico.
15. **COMUNICACIÓN CUÁNTICA:** Es el tipo de comunicación entre dos sistemas que se da debido a la computación cuántica.
16. **CUERPOS OSCUROS:** Es una entidad u objeto que consume toda la energía que reside cerca de él; este absorbe toda la luz igual.
17. **EFECTO FOTOELÉCTRICO:** Es la emisión de electrones desde un material cuando ha este se le introduce una radiación electromagnética.
18. **ENCRIPTACIÓN:** Es una técnica o procedimiento que se utiliza para, por medio de una clave, cambiar la lingüística o sentido de un mensaje.
19. **ENTRELAZAMIENTO:** Es una característica cuántica que produce dependencia entre dos fotones, o en el caso de la computación cuántica, e bit cuántico; cuando uno cambia de estado o de giro, produce el cambio instantáneo del otro, independientemente de su distancia.
20. **ESFERA DE BLOCH:** Es la representación grafica de un sistema cuántico de dos niveles.
21. **ESPACIO DE HILBERT:** Es una representación generalizada de un espacio euclídeo, el cual en el caso de la computación cuántica tiene un rango de 0 a dos pis.
22. **EXPERIMENTO DE YOUNG:** Es un experimento que llevo a cabo Thomas Young en el cual una pistola de fotones dispara fotones a través de una pared con dos agujeros y detrás de esta reside otra pared recibiendo el choque de los fotones.
23. **FOTÓN:** Es una partícula elemental que se da debido a la radiación electromagnética.

24. **FPGA:** Es un circuito integrado cuyas interconexiones pueden ser reconfiguradas por un usuario por manufactura.
25. **FUNCIÓN DE EULER:** Es una función matemática que relaciona un numero n con distintos números que son coprimos, números que al multiplicarse dan como respuesta n.
26. **FUNCIÓN DE ONDA:** Es una entidad matemática que describe la representación grafica de un espectro de luz o fotones.
27. **INTERACCIONES DE VECINO:** Es una representación grafica del comportamiento de partículas subatómicas.
28. **LA PARADOJA EPR:** Es el concepto de tener una dependencia entre dos partículas independientemente de su distancia, la cual también describe que la teoría de la mecánica cuántica está incompleta.
29. **LEY DE DESPLAZAMIENTO:** Es una ley física que describe que hay una relación inversamente proporcional entre el pico de onda de un cuerpo oscuro y su temperatura.
30. **MAQUINA DETERMINISTA DE TURING:** Es una maquina que por medio de un conjunto de reglas interpreta de una manera u otra el flujo de una cinta con agujeros que pasa internamente por ella.
31. **MECÁNICA CUÁNTICA:** Es el estudio físico de las partículas en una escala subatómica.
32. **MEMORIA DDR:** Es un tipo de memoria RAM.
33. **METODOLOGÍA INCREMENTAL:** Es una metodología la cual se rige por distintos ciclos en los cuales al ser finalizados cada uno el siguiente ciclo tiene como objetivo mejorar o poner a prueba de una manera más compleja al sistema que se está diseñando o fabricando.
34. **MICROPROCESADORES:** Es un conjunto de circuitos integrados diseñado para hacer funciones mas complejas que estos a mayores velocidades, y con mayor densidad de transistores y otros dispositivos.
35. **MOTOR DE SZILARD:** Es un motor con una cámara vacía con un separador a la mitad de esta, y con un pistón a cada extremo de la estructura; cabe destacar que este nunca fue fabricado y es más que todo utilizado para modelar un sistema clásico binario.

36. **NOTACIÓN DE DIRAC:** Es una notación que permite representar de dos formas distintas un vector, ya sea como una columna representado por un ket, o como una línea representado por un bra
37. **OPERADORES UNITARIOS:** Son entidades matemáticas que se utilizan mayormente para no modificar la magnitud de un numero o sistema, pero si su posición o visualización.
38. **PARALELISMO CUÁNTICO:** Es la idea o concepto que se maneja de que la computación cuántica puede realizar todo al mismo tiempo.
39. **PLD:** Es un componente electrónico, un circuito integrado, diseñado para ser reprogramado.
40. **PROM:** Es un tipo de memoria que solo puede ser utilizado una vez debido a que funciona con fusibles y estos al quemarse solo puede grabar la primera información que se guardó.
41. **RESONADORES:** Es un dispositivo capaz de causar resonancia, lo cual es poder causar la vibración de un objeto a distintas frecuencias y amplitudes.
42. **SEMICONDUCTORES DE UNIÓN BIPOLAR:** Son dispositivos de estado solido que permiten el flujo de corriente entre su colector y emisor debido a una diferencia de potencial en su base.
43. **SUPERPOSICIÓN:** Es una característica cuántica que describe que una entidad cuántica, como el fotón o el bit cuántico, pueda ser dos estados simultáneamente, esta teniendo posibilidades distintas de caer en cualquiera de sus estados.
44. **TIEMPO POLINÓMICO:** Cuando el tiempo de ejecución de un algoritmo se da en menor tiempo de algún valor calculado debido a una función polinomial.
45. **TRANSFORMADA CUÁNTICA DE FOURIER:** Es una compuerta cuántica que es la equivalencia cuántica de la transformada de Fourier discreta.
46. **VERILOG:** Es un tipo de programación descriptiva de hardware; con una diferente sintaxis que VHDL.
47. **VHDL:** Es un tipo de programación descriptiva de hardware; con una diferente sintaxis que Verilog.

REFERENCIAS

Abhishek Pandey, R. V. (2015). *Quantum computing for big data analysis*. 98-104. 14(43).

Abramowitz, M. (2015). *Electromagnetic Radiation*. Olympus. https://www.olympus-lifescience.com.cn/en/microscope-resource/primer/java/electromagnetic/

Barkalov, A., & Titarenko, L. (2008). *Logic Syntheis for Compositional Microprogram Control Units* (22.ª ed.). Springer.

Bennett, C. H., & DiVincenzo, D. P. (2000). *Quantum information and Computation* (Vol. 404). Nature.

Casagrande, D. G. (1999). *Information as Verb: Re-conceptualizing Information for Cognitive and Ecological Models*. *3*, 4-13.

Church, A. (1936). *An unsolvable problem of elemtary number theory*. *58*, 345-363.

Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). *Introduction to Algorithms* (3.ª ed.). Massachusetts Institute of Technology.

Deutsch, D. (1985). *Quantum Theory, the Church-Turing Principle and the Universal Quantum Computer*. *400*(1818), 97-115.

Deutsch, D. (1989). *Quantum computational network*. 73-90.

Deutsch, D., & Jozsa, R. (1992). *Rapid solution of problems by quantum computation*. 553-558.

Dirac, P. A. M. (1981). *The Principles of Quantum Mechanics* (4.ª ed.). Oxford University Press.

Einstein, A. (1905). *Concerning an Heuristic Point of View Toward the Emission and Trnasformation of Light*. *33*.

Einstein, A., Podolsky, B., & Rosen, N. (1935). *Can Quantum-Mechanical Description of Physical Reality Be Considered Complete? 47*, 777-780.

Feynman, R. P. (1982). *Simulating Physics with Computers*. *21*, 22.

Griffiths, D. J. (1995). *Introduction to Quantum Mechanics* (2.ª ed., Vol. 1). Prentice Hall.

Gruska, J. (2000). *Quantum Computing*. McGraw-Hill Book Co Ltd.

Jozsa, R., & Linden, N. (2002). *On the role of entanglement in quantum computational speed´-up*.

Karafylidis, I. G. (2005). *Quantum Computer Simulator based on the Circuit Model of Quantum Computation*. *52*, 1590-1596.

Karnaugh, M. (1953). *The Map Method for Synthesis of Combinational Logic Circuits*. *72*(5), 593-599.

Kent, A. P., Munro, W. J., Spiller, T. P., & Beausoleil, R. G. (2003). *Quantum Cryptography*. United States Patent; US 7,983,422 B2.

Khalid, A. U. (2005). *FPGA Emulation of Quantum Circuits*. McGrill University.

Ladd, T. D., Jelezko, F., Laflamme, R., Nakamura, Y., Monroe, C., & O'Brien, J. L. (2010). *Quantum Computing*. 45-53.

Lu, C.-Y., Browne, D. E., Yang, T., & Pan, J.-W. (2007). *Demosntration of a Complide Version of Shor's Quantum Factoring Algorithm Using Photonic Qubits*. 5.

Marinescu, D. C., & Marinescu, G. M. (2012). *Classical and Quantum Information*. Elsevier.

McCulloch, M. (2015, octubre 9). *Physics from the edge*. BlogSpot. http://physicsfromtheedge.blogspot.com/2015/10/mihsc-from-bit.html

Mills, H. D. (1980). *The management of software engineering. Part 1: Principles of software engineering*. *19*(4), 414-477.

Moore, G. E. (1965, abril 19). The experts look ahead. *Cramming more components onto integrated circuits*, *38*(8).

Mujtaba, H. (2019, octubre 22). *AMD 2nd Gen EPYC Rome Processors Feature A Gargantuan 39.54 Billion Transistors, IO Die Pictured in Detail*. wccftech.

Nielsen, M. A., & Chuang, I. L. (2010). *Quantum Computation and Quantum Information* (10mo Aniversario, Vol. 7). Cambridge University Press.

Numato Lab. (s. f.). Mimas V2 Spartan 6 FPGA Development Board With DDR SDRAM. https://numato.com/product/mimas-v2-spartan-6-fpga-development-board-with-ddr-sdram

Ortiz, M. (2017). *Modelo Incremental*. Ingenieria de Software. http://isw-udistrital.blogspot.com/2012/09/ingenieria-de-software-i.html

Planck, M. (1901). *On the Law of Distribution of Energy in the Normal Spectrum*. *4*, 553-564.

Schrödinger, E. (1935). *Die gegenwiirtige Situation in der Quantenmechanik*. *124*(5), 323-338.

Shannon, C. E. (1948). *A Mathematical Theory of Communication*. *27*(3), 379-423.

Shannon, C. E. (1956). A Universal Turing Machine with two tates. En *Automata Studies: Annals of Mathematic Studies* (Vol. 34, pp. 157-164). Princeton University Press.

Silva, A. (2018). *Emulador de algoritmos cuánticos en FPGA utilizando herramientas de diseño de alto nivel*. Universidad Nacional de Mar del Plata.

Singhal, J. (2016, agosto). *An overview of Quantum Computing*. ReaserchGate. https://www.researchgate.net/publication/309653719_An_overview_of_Quantum_Computing

Sysoev, S. S. (2019). *Introduction to quantum computer. Notes Week 2*.

Turing, A. M. (1937). On Computable Numbers, with an Application to the Entsheidungsproblem. *Proceedings of the London Mathematical Society*, *42*, 230-265.

Vedral, V., & Plenio, M. B. (1998). Basics of quantum computation. En *Progress in Quantum Electronics* (Vol. 22, pp. 1-39). Elsevier. https://www.sciencedirect.com/science/article/abs/pii/S0079672798000044

WayBackMachine. (2015). History of FPGAs.

Young, T. (1802). *On The Theory of Light and Colours*. 12-39.