Introducción al Desarrollo de Software Cuántico

José Manuel García Alonso

UNIVERSIDAD DE EXTREMADURA

jgaralo@unex.es

ELI – III Escuela Latinoamericana de Informática

Software Engineering Group
QUERCUS

SPILab

PROGRAMA DEL CURSO

CONTENIDO	DÍA
Introducción a la Programación Cuántica	Martes
Primitivas Cuánticas: Estructura de un programa cuántico	Miércoles
Aplicaciones: Algunos Algoritmos	Jueves
Servicios Cuánticos	Viernes

Para la aprobación y/o certificado de asistencia al curso los asistentes deberán concurrir al 80% de las clases dictadas (4 clases).
Para la aprobación del curso se realizará un trabajo final de unas 15 hs. extras en tema a acordar con el docente durante la ELI, a entregar en las 2 semanas posteriores al dictado del curso (propuesta para uniformizar las entregas).
El mecanismo de entrega se informará más adelante.

Introducción al Desarrollo de Software Cuántico

Juan Manuel Murillo Rodríguez
José Manuel García Alonso

UNIVERSIDAD DE EXTREMADURA

juanmamu@unex.es – jgaralo@unex.es

ELI – III Escuela Latinoamericana de Informática

Primitivas Cuánticas

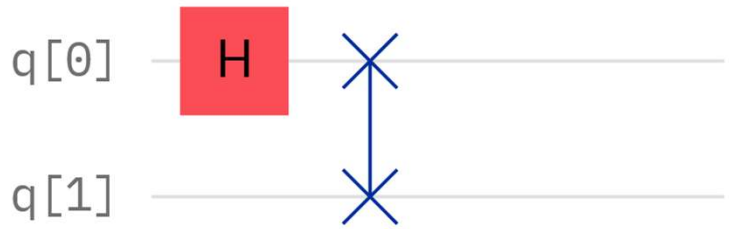
EQUIVALENCIA ENTRE PUERTAS

Hay una serie de puertas cuánticas que pueden implementarse como una combinación de puertas más simples.

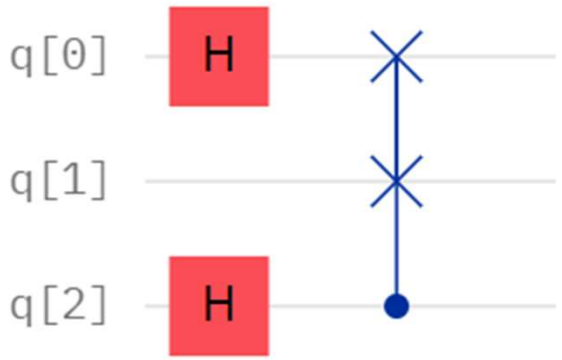
Veamos algunos ejemplos...

EQUIVALENCIA ENTRE PUERTAS

- SWAP

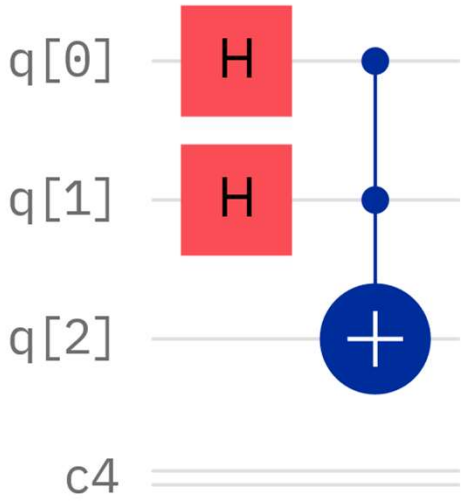


- CSWAP

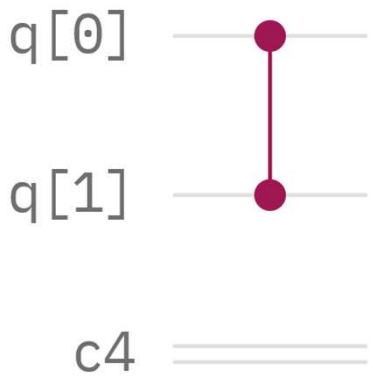


EQUIVALENCIA ENTRE PUERTAS

- Toffoli



- CZ (CPHASE(180))



Aritmética y Lógica (A&L): Objetivos

- Definir sumas, restas y multiplicaciones en superposición utilizando puertas cuánticas.
- Implementar el incremento, decrement y adición en simulador cuántico **(IBM Quantum Composer).**
- Utilizar números enteros negativos.
- Adaptar la lógica booleana a las operaciones de la QPU.

A&L: Enfoque de alto nivel

En programación clásica no usamos puertas lógicas individuales para escribir programas

Es el compilador y la CPU los encargados de convertir nuestros programas en las "puertas" necesarias.

Al igual que la lógica clásica puede construirse a partir de puertas *NAND* (una sola puerta que realiza *NOT (a AND b)*), las operaciones cuánticas con números enteros pueden construirse a partir de las operaciones elementales de la QPU.

A&L: Reversibilidad

A diferencia de muchas operaciones lógicas convencionales, las operaciones básicas de QPU son **reversibles**. (Operaciones con Matrices Unitarias)

La forma más sencilla de implementar cualquier circuito convencional en nuestra QPU **es sustituirlo por un circuito convencional equivalente**, que solo utilice operaciones reversibles, como las puertas Toffoli.

Podemos implementarlo de forma virtual sobre un registro cuántico (simulación)

A&L: Aritmética Tradicional vs Cuántica

- La lógica convencional tiene muchos enfoques bien optimizados para realizar operaciones aritméticas.

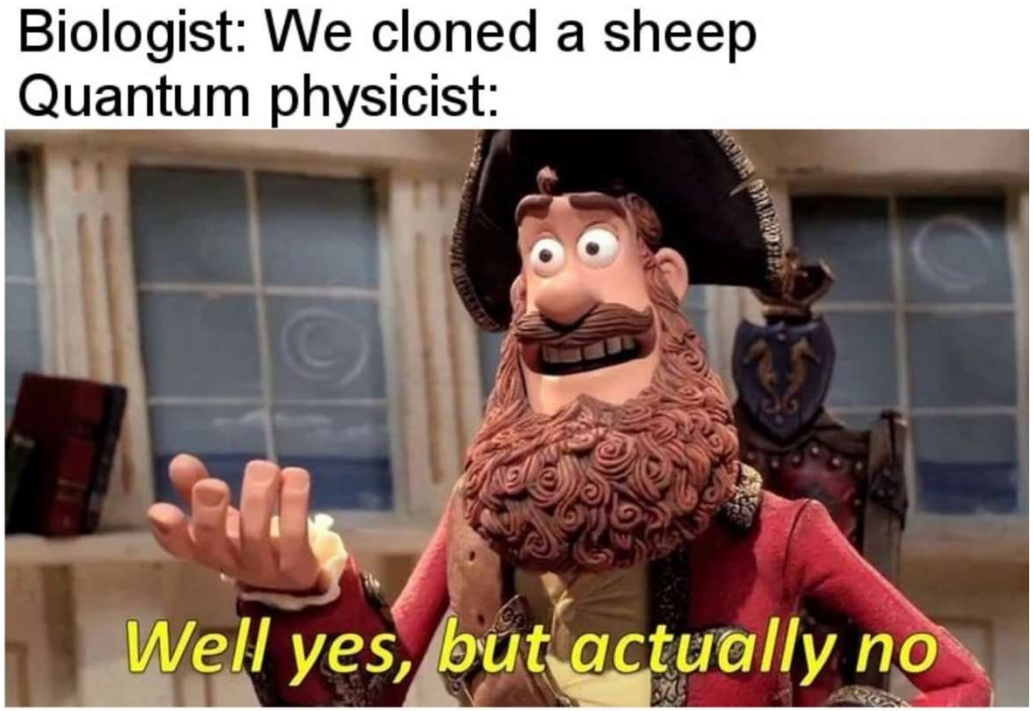
¿Por qué no podemos sustituir bits por qubits y utilizarlos en nuestra QPU?

- Tenemos registros de entrada que están en superposición.
- Queremos que las operaciones aritméticas cuánticas afecten a todos los valores de la superposición.

A&L: Aritmética Tradicional vs Cuántica

¡ Los qubits no se pueden copiar !

(Non-cloning Theorem)



A&L: Aritmética Tradicional vs Cuántica

¡ Los qubits no se pueden copiar !

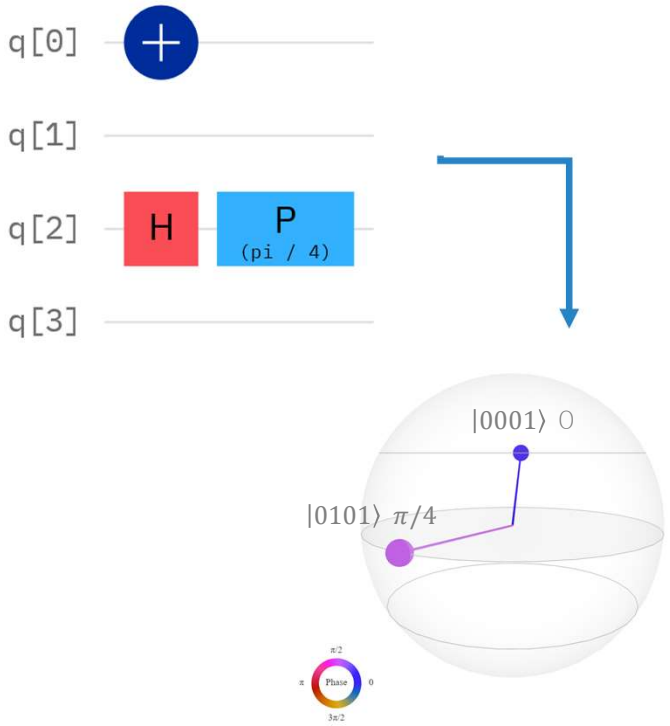
(Non-cloning Theorem)

- Una QPU puede alterar los qubits utilizando instrucciones de intercambio, pero ninguna QPU implementa una instrucción de COPIAR.
- En consecuencia, el operador = no puede utilizarse para asignar un valor de un qubit a otro.
- Solo podemos actualizar valores de registro como ++, --, x2, etc.


hands
on

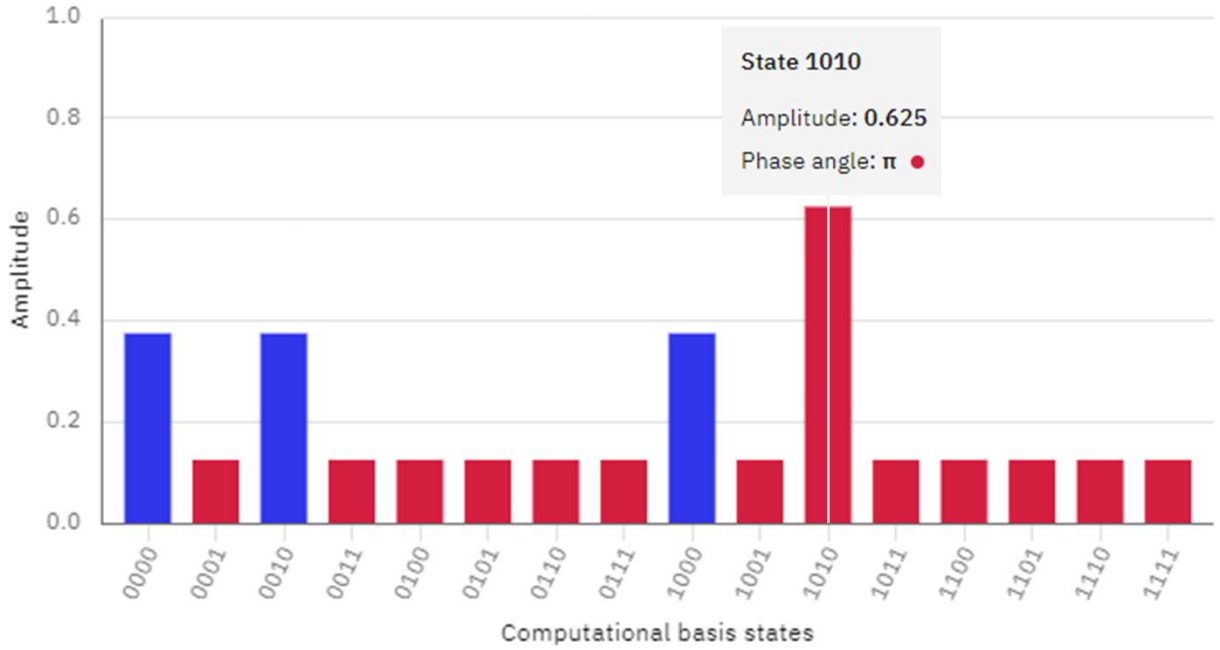
A&L: Representación Q-sphere

- Q-sphere representa el estado de un sistema de uno o más qubits, al asociar cada estado con un punto en la superficie de una esfera. Un nodo es visible en cada punto.
- El color del nodo indica la rotación de fase cuántica y el radio su probabilidad.
- Los nodos se disponen en la esfera de modo que el estado base con **todos ceros** (por ejemplo $|0000\rangle$) está en su **polo norte**, y el estado base con **todos unos** (por ejemplo $|1111\rangle$) está en su **polo sur.**



A&L: Representación Statevector

- Amplitud es el tamaño de la barra
- Fase es el color

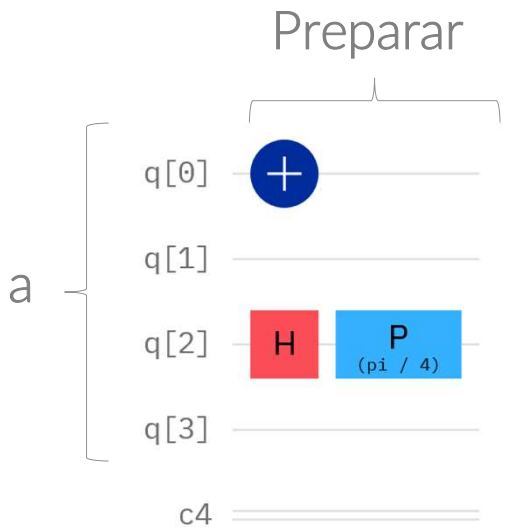


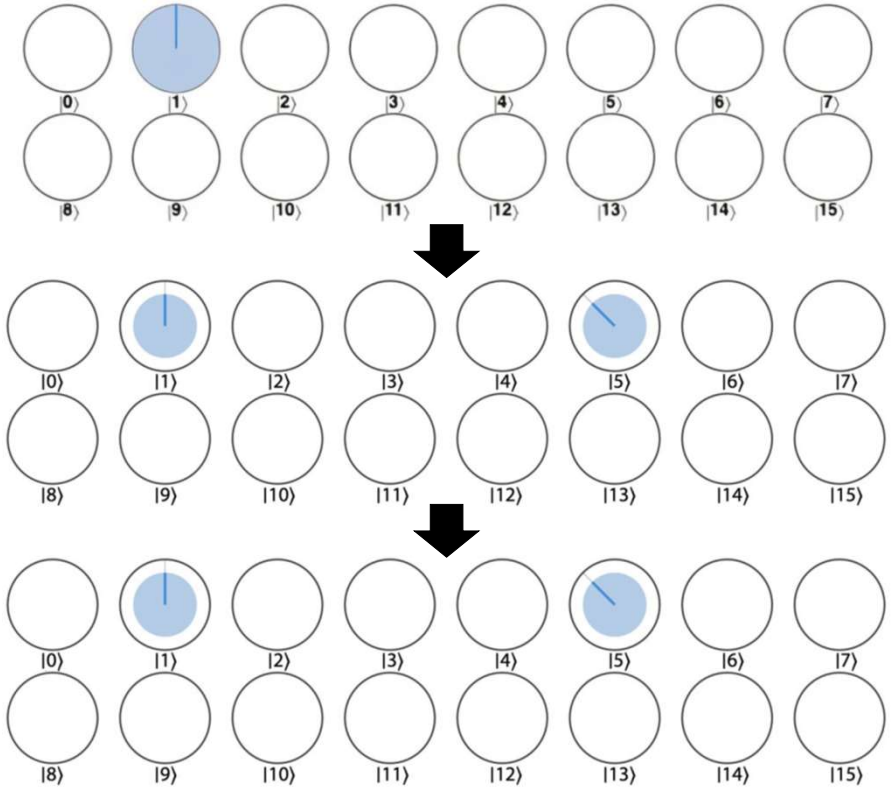
A&L: Operador Incremento

Ejercicio: a++

- **Paso 1. Preparación inicial**

  Creación de la configuración inicial



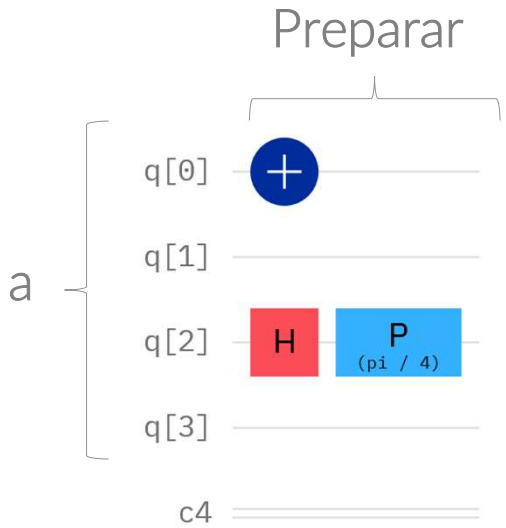


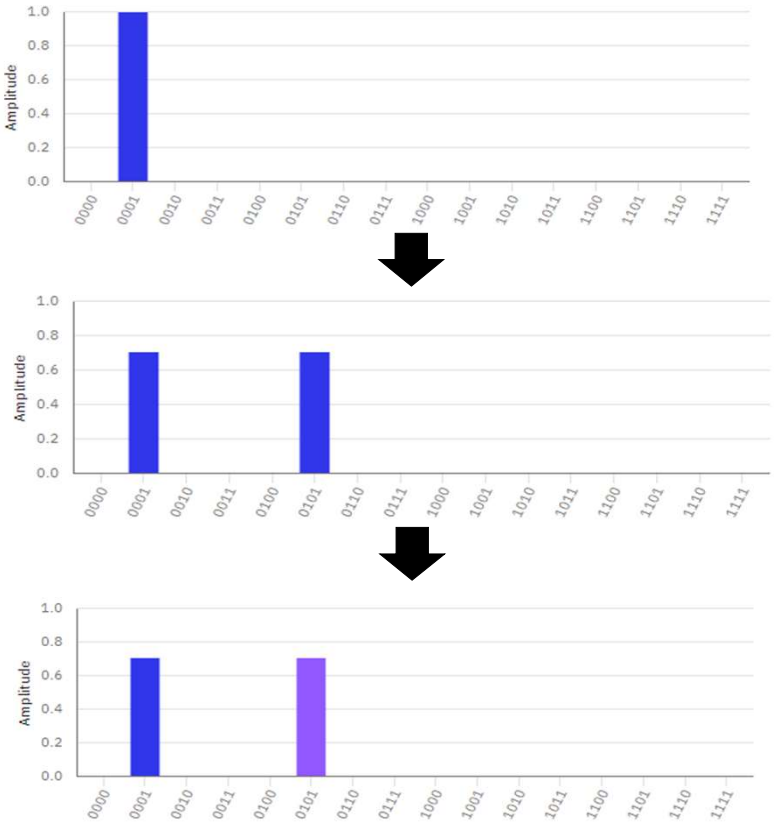
A&L: Operador Incremento

Ejercicio: a++

Paso 1. Preparación inicial

- Creación de la configuración inicial





A&L: Operador Incremento

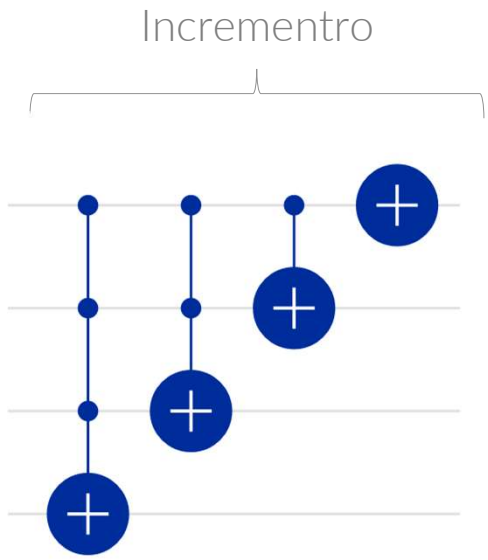
Ejercicio: a++

Paso 2: Operación de incremento

- La operación ++ comienza utilizando una puerta CNOT de tres condiciones:

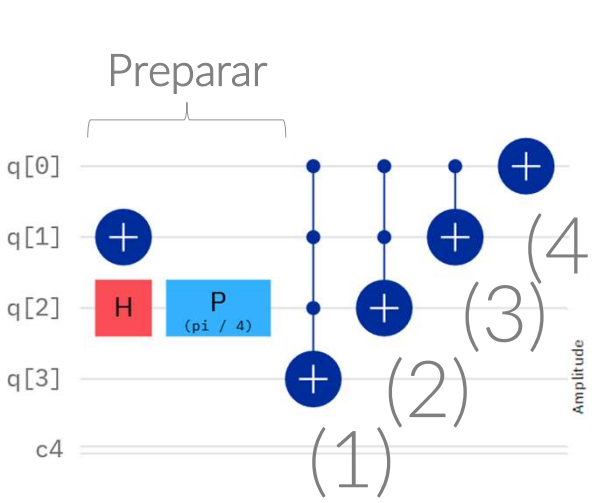
  Esta se aplica si todos los bits de menor nivel tienen valor 1, entonces altera el valor del bit de mayor nivel **(operación aritmética de acarreo (*carry*), llevarse en primaria)**

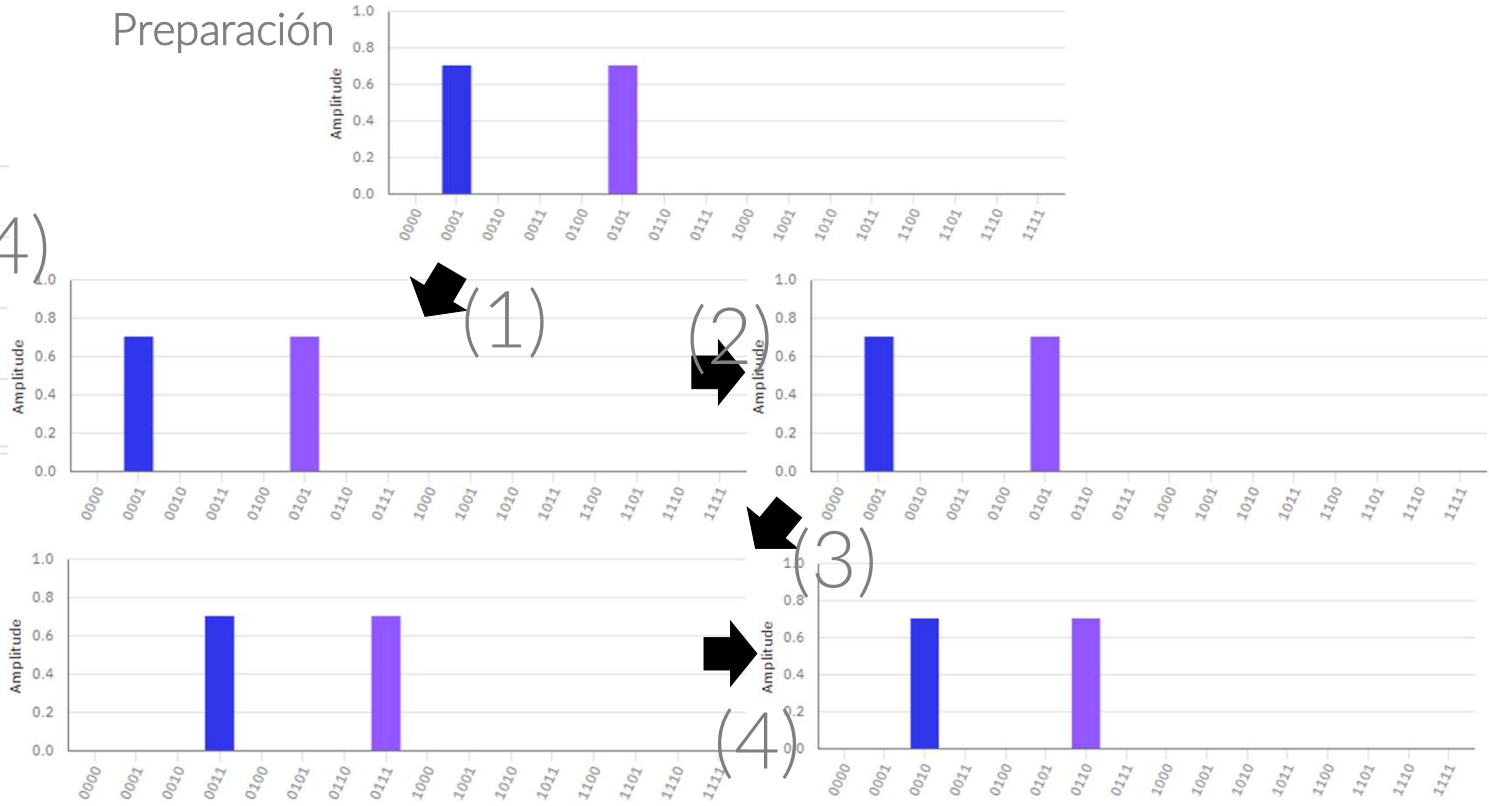
- A continuación, repetimos el proceso, lo que resulta en una operación completa de “suma y acarreo” en todos los qubits, realizada solo con puertas CNOT y NOT.



A&L: Operador Incremento

Ejercicio: a++



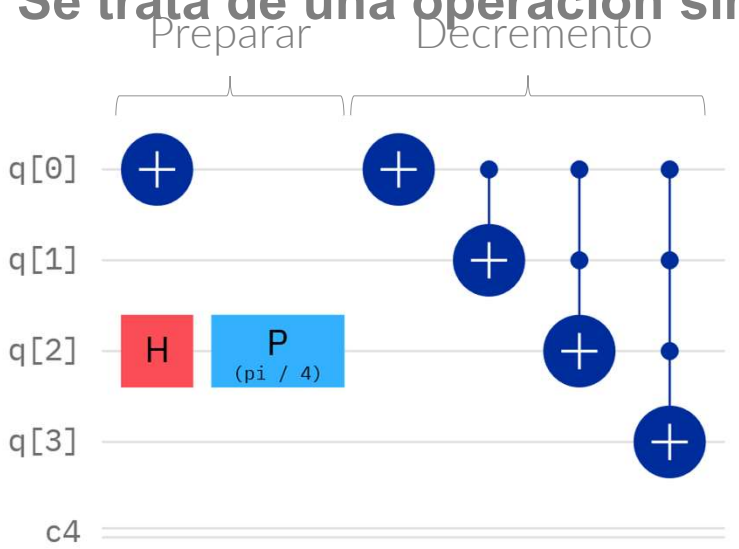


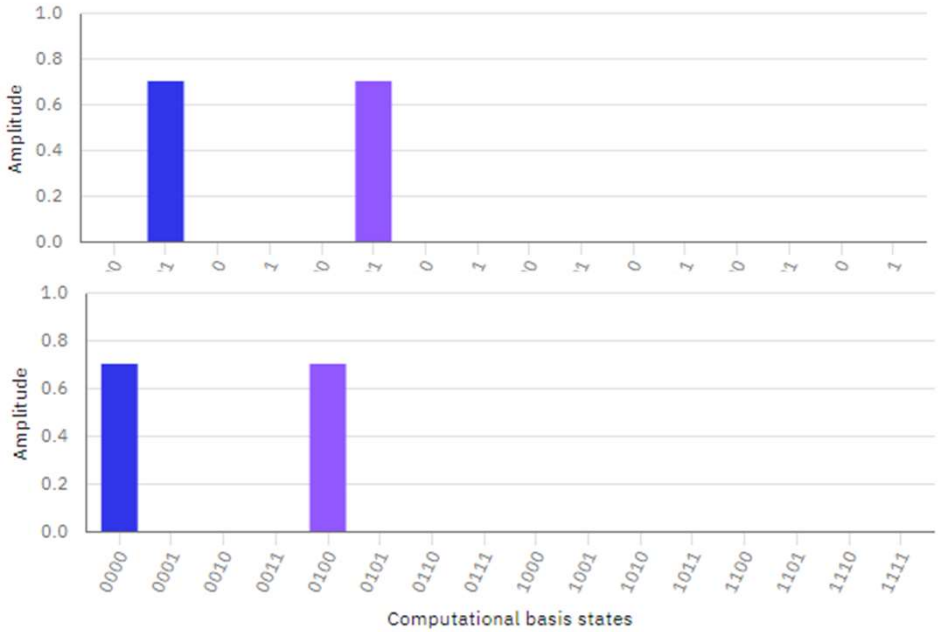
A&L: Operador Decremento

Ejercicio: a--

Vamos a definir el circuito para realizar la operación de decremento.

Se trata de una operación simétrica a a++ .

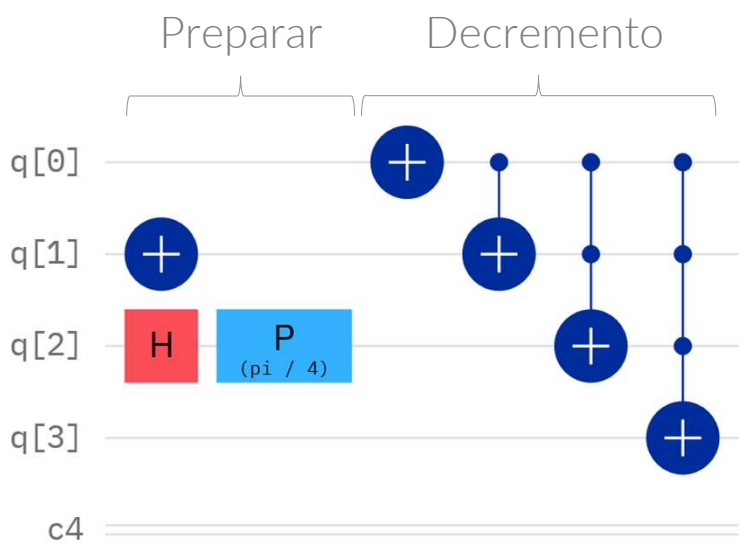


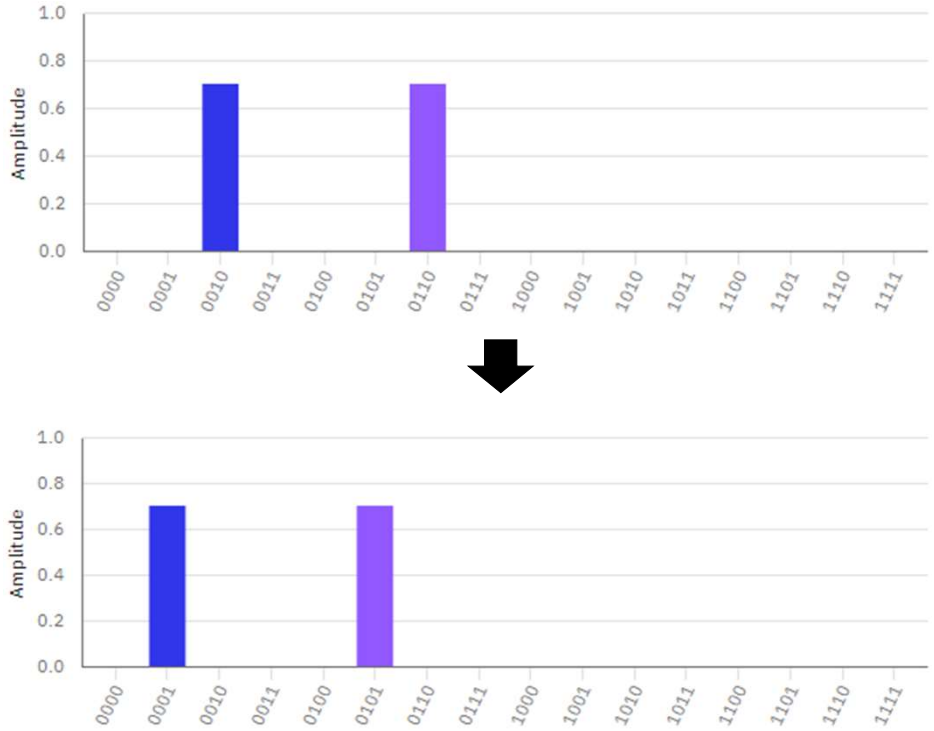


A&L: Operador Decremento

Ejercicio: a--

Qué ocurre si inicializamos q[1] a 1

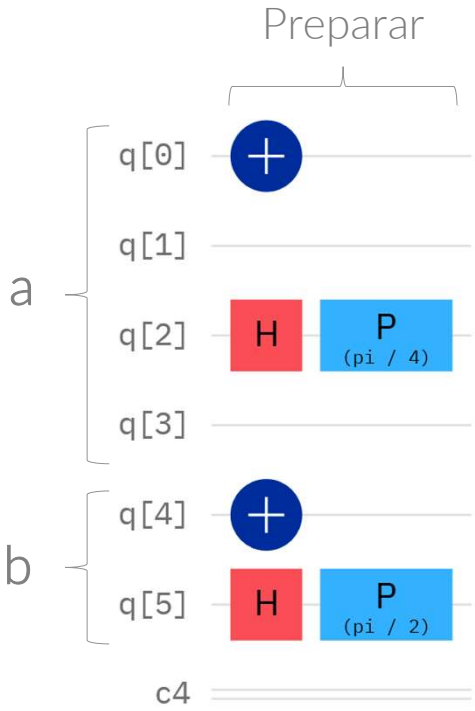


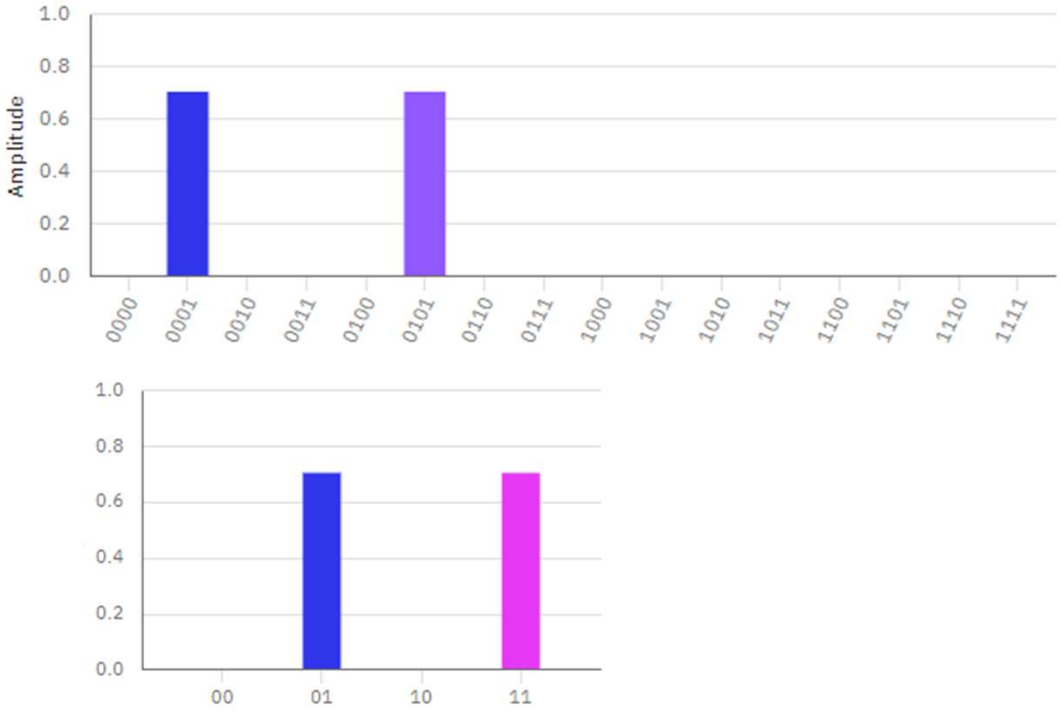


A&L: Sumar Enteros

Ejercicio: a += b

Paso 1. Preparación inicial





A&L: Sumar Enteros

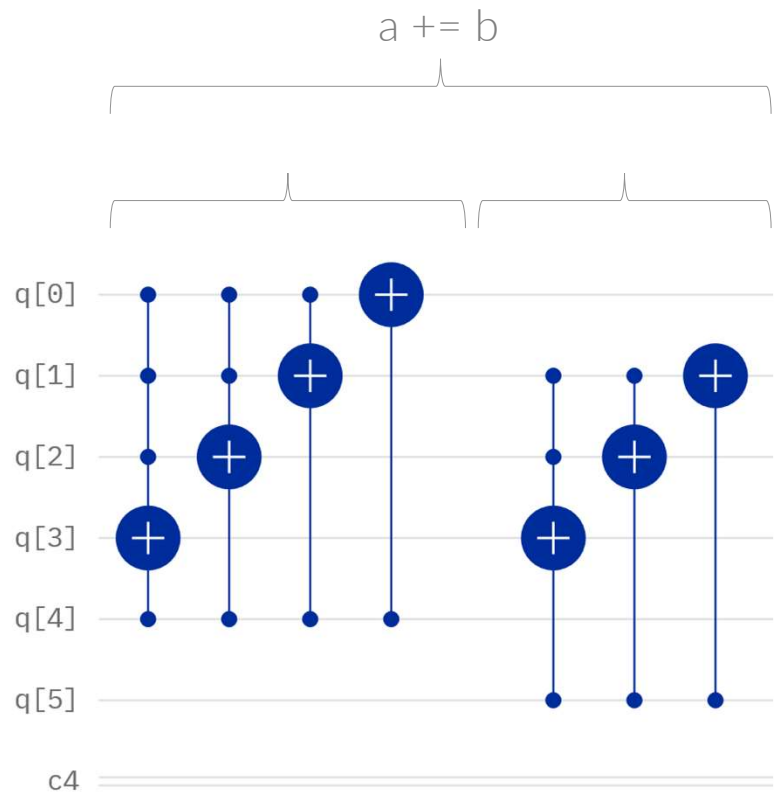
Paso 2. Operaciones de incremento aplicadas a a

- Las puertas a utilizar son simplemente las operaciones de incremento de enteros aplicadas a **a**.

  Pero se realiza en función de los qubits correspondientes de **b**

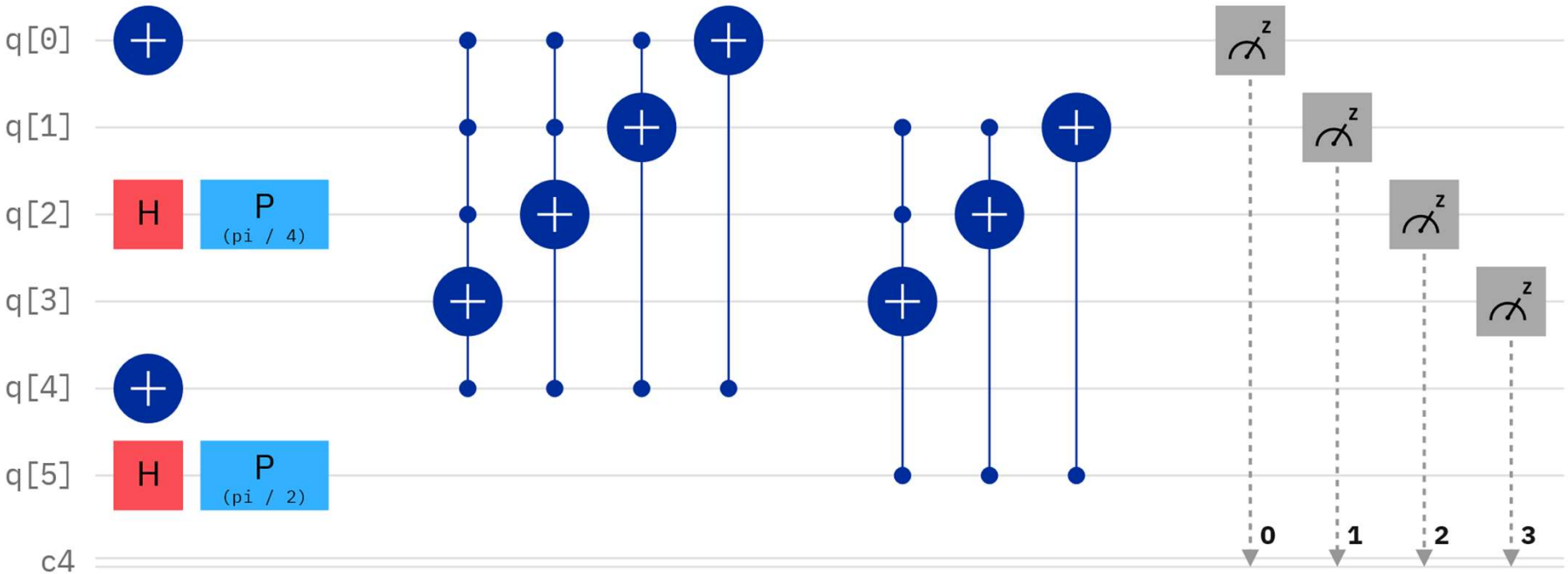
  Esto permite que los valores en **b**, incluso en superposición, determinen el resultado de la suma.

Ejercicio: a += b



A&L: Sumar Enteros

Ejercicio: a += b



A&L: Números enteros negativos

- **Para conseguir un número entero negativo se realiza la negación en complemento a dos, es decir, voltear todos los qubits y sumar 1.**
- Para un número determinado de bits, simplemente asociamos la mitad de los valores con números negativos y la otra mitad con números positivos.
- Por ejemplo, un registro de tres bits nos permite representar los números enteros:

-4, -3, -2, -1, 0, 1, 2, 3

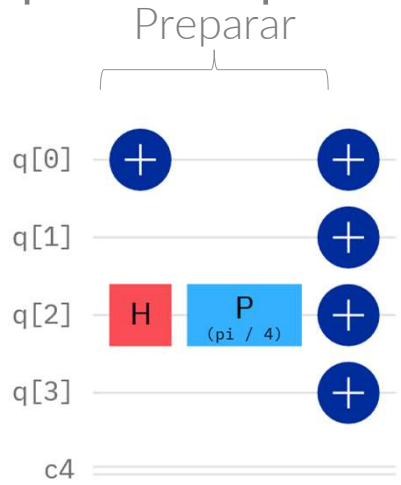
0	**1**	**2**	**3**	**–4**	**–3**	**–2**	**–1**
000	001	010	011	100	101	110	111

A&L: Números enteros negativos

Ejercicio a=-a

Paso 1. Preparación inicial

- Preparación inicial mediante puerta Hadamard.
- Para negar un número en complemento a dos, primero invertimos todos los qubits con puertas NOT.

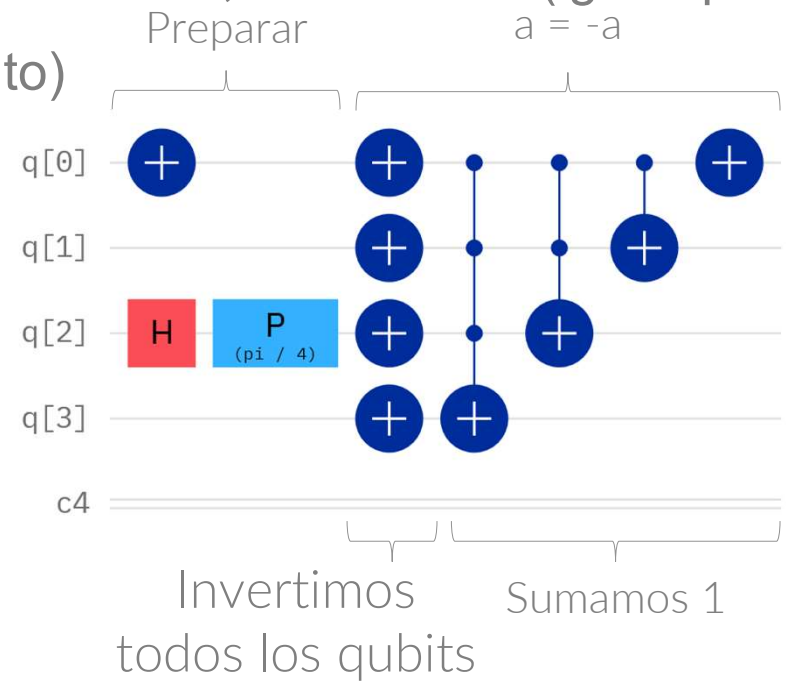


A&L: Números enteros negativos

Ejercicio a=-a

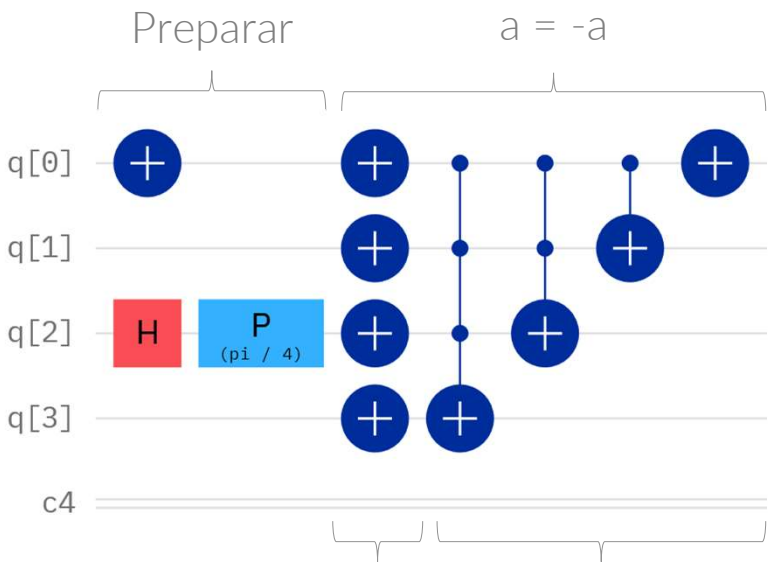
Paso 2. Incremento

- Una vez invertidos, sumamos 1 (igual que en el ejemplo visto en el incremento)



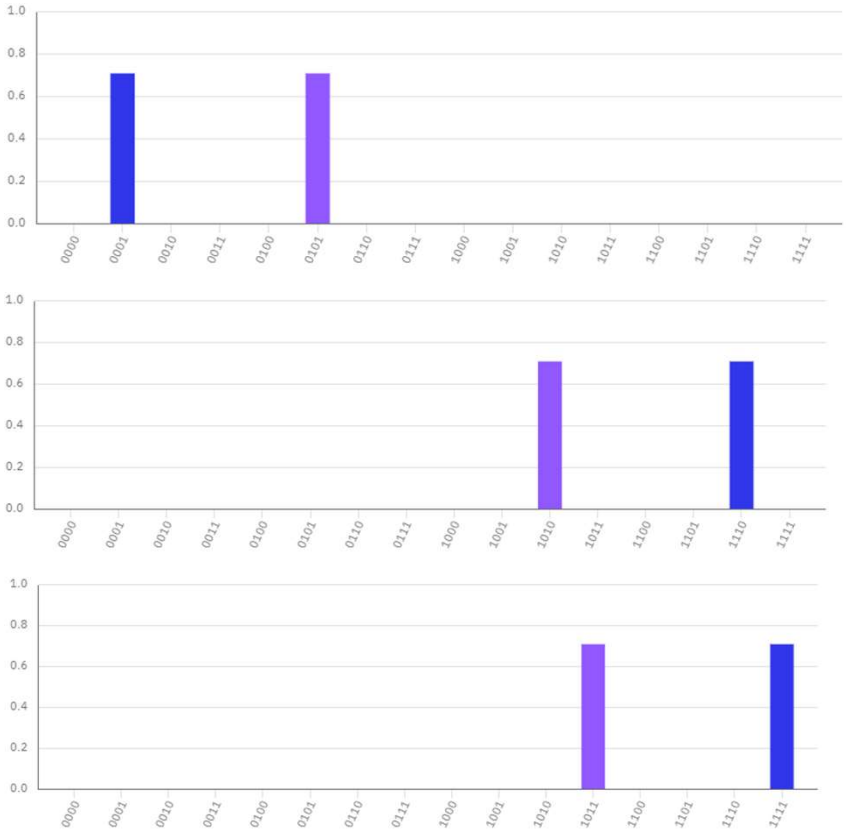
A&L: Números enteros negativos

Ejercicio a=-a



Invertimos todos los qubits

Sumamos 1

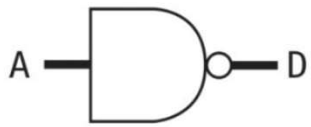


A&L: Adaptación de la lógica booleana

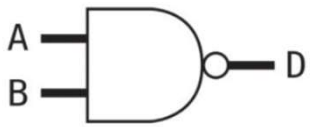
Lógica clásica: NAND

En lógica digital clásica, algunas puertas lógicas pueden usarse para construir todas las demás:

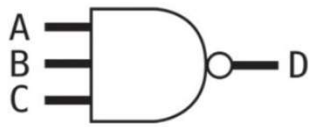
- **Por ejemplo, si solo tienes una puerta NAND, puedes usarla para crear AND, OR, NOT, y XOR, que pueden combinarse en cualquier función lógica que desees.**
- Hay que tener en cuenta que las puertas NAND pueden tener cualquier número de entradas (con una dola entrada, NAND realiza un NOT).



D = **NOT** (A)



D = **NOT** (A **AND** B)

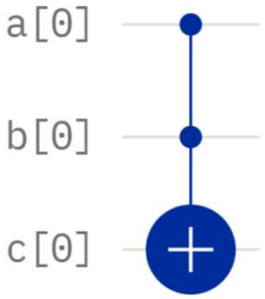


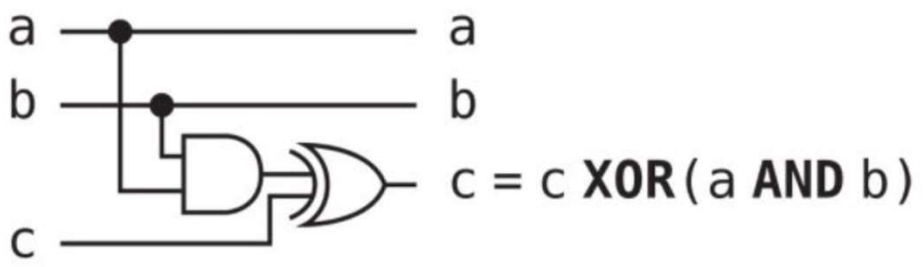
D = **NOT** (A **AND** B **AND** C)

A&L: Adaptación de la lógica booleana

Lógica Cuántica: CNOT y Toffoli

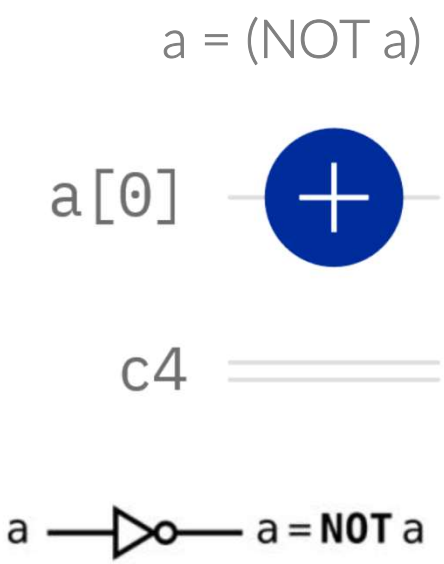
- En la computación cuántica podemos empezar de forma similar con puertas versátiles y construir nuestra lógica digital cuántica a partir de ellas.
- Para ello, **utilizaremos puertas CNOT multi-controladas y puertas Toffoli.**
- Al igual que NAND, **podemos variar el número de entradas para ampliar la lógica.**

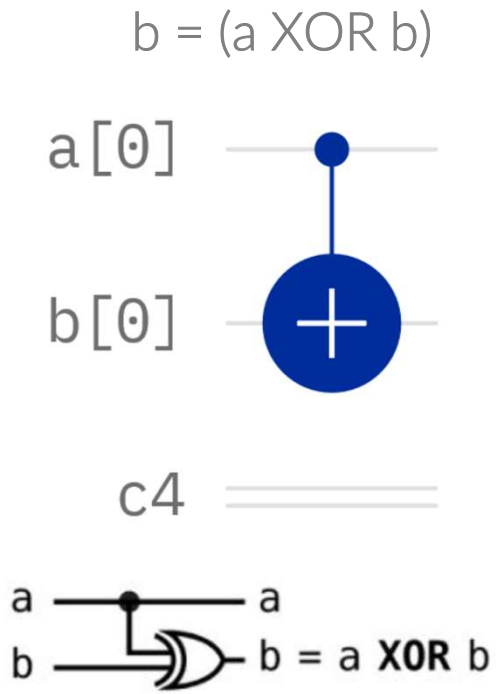




A&L: Adaptación de la lógica booleana

Ejemplos de puertas lógicas cuánticas

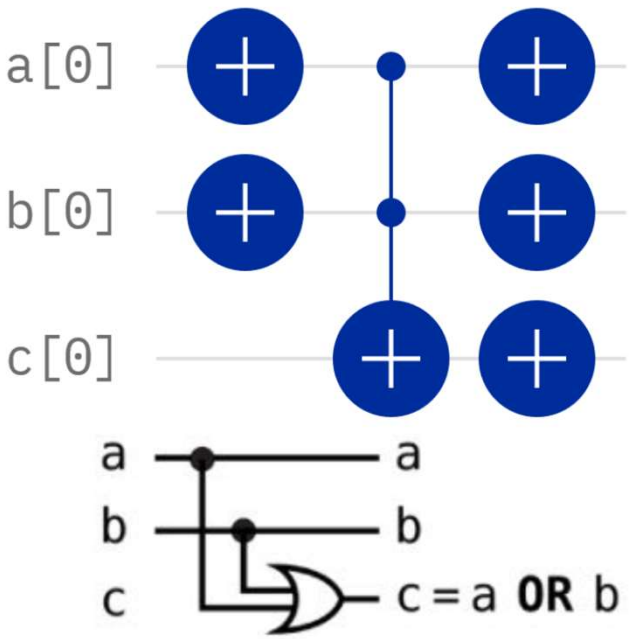




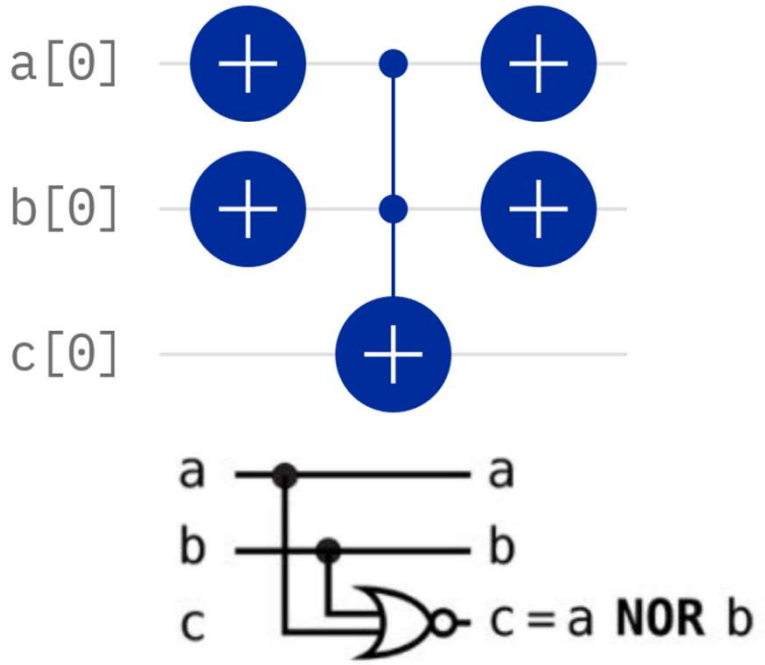
A&L: Adaptación de la lógica booleana

Ejemplos de puertas lógicas cuánticas

c = (a OR b)

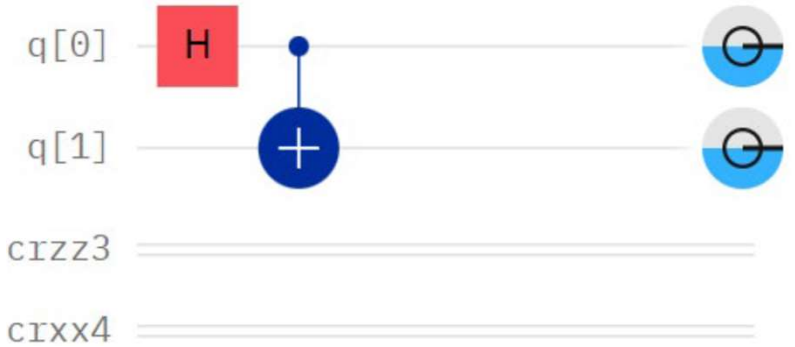


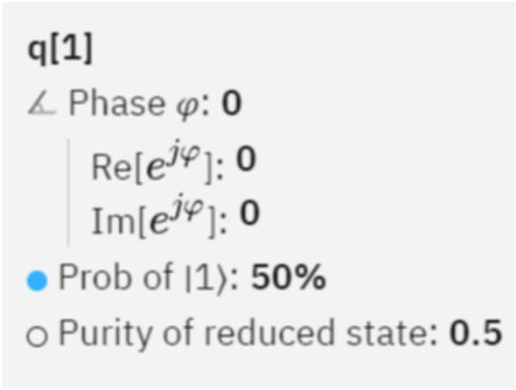
c = (a NOR b)



Transición de Fase en IBM Composer

- En física, una transición de fase cuántica (QPT) es una transición de fase entre diferentes fases cuánticas.
- Las transiciones de fase cuántica se producen como resultado de la interacción de las fases del estado básico.
- En IBM Quantum Composer, el disco de fase en el extremo de cada qubit proporciona el estado de cada qubit al final del cálculo.





Transición de Fase en IBM Composer

Puertas de cambio de fase en IBM Composer

T	T gate	$\varphi \rightarrow \varphi + \pi/4$
S	S gate	$\varphi \rightarrow \varphi + \pi/2$
Z	Z gate	$\varphi \rightarrow \varphi + \pi$

Puerta general de cambio de fase:

P	$P(\phi)$ gate	$\varphi \rightarrow \varphi + \phi,$

Transición de Fase en IBM Composer

Puertas de cambio de fase en IBM Composer

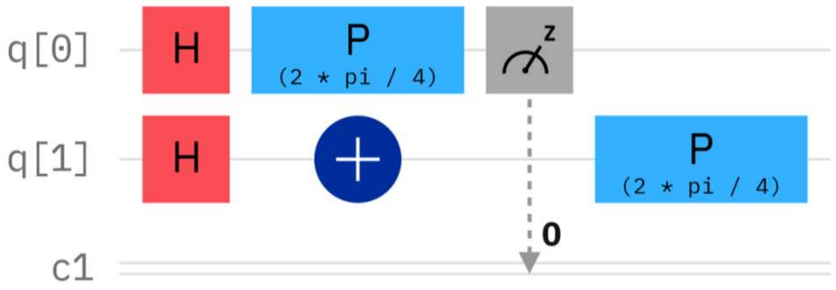
Las inversas de las puertas T y S respectivamente:

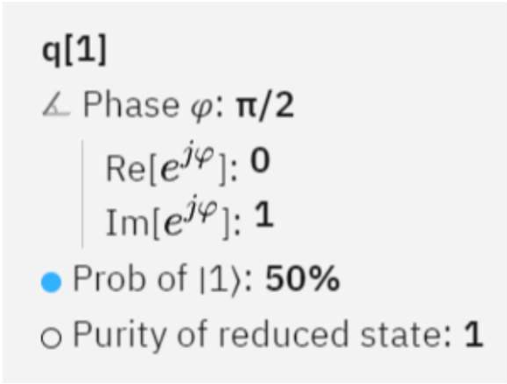
$T^\dagger$	$T^\dagger$ gate	$\varphi \rightarrow \varphi - \pi/4$
$S^\dagger$	$S^\dagger$ gate	$\varphi \rightarrow \varphi - \pi/2$

Para agregar una fase cuántica a un estado, en el circuito cuántico consiste en aplicar primero una puerta Hadamard y, a continuación, añadir una o más de las puertas de cambio de fase.

Transición de Fase en IBM Composer

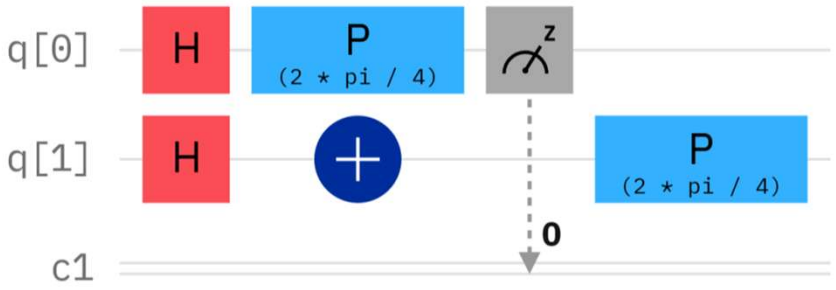
Ejemplo de visualización de transición de fase con la puerta general de cambio de fase

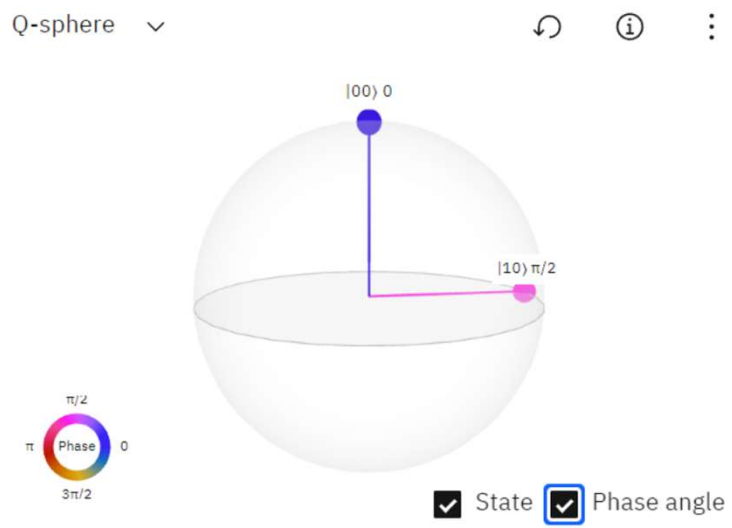




Transición de Fase en IBM Composer

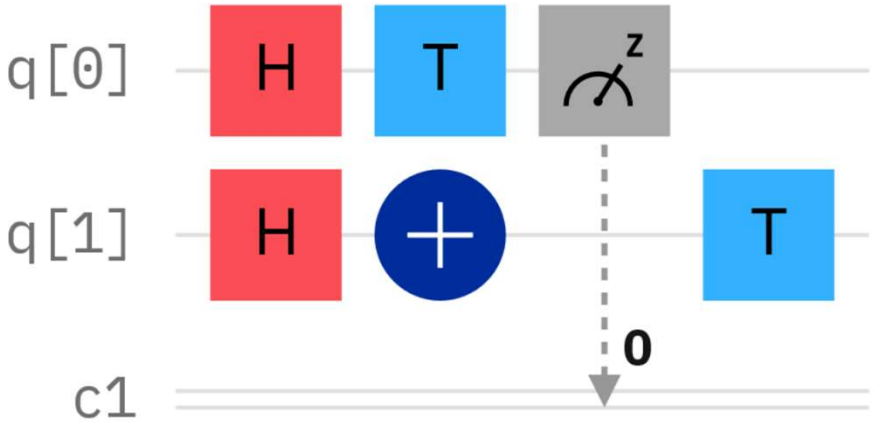
Ejemplo de visualización de transición de fase con la puerta general de cambio de fase

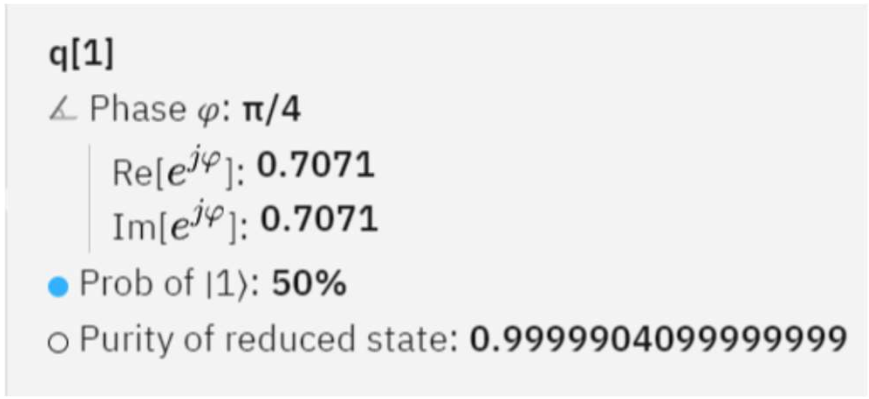




Transición de Fase en IBM Composer

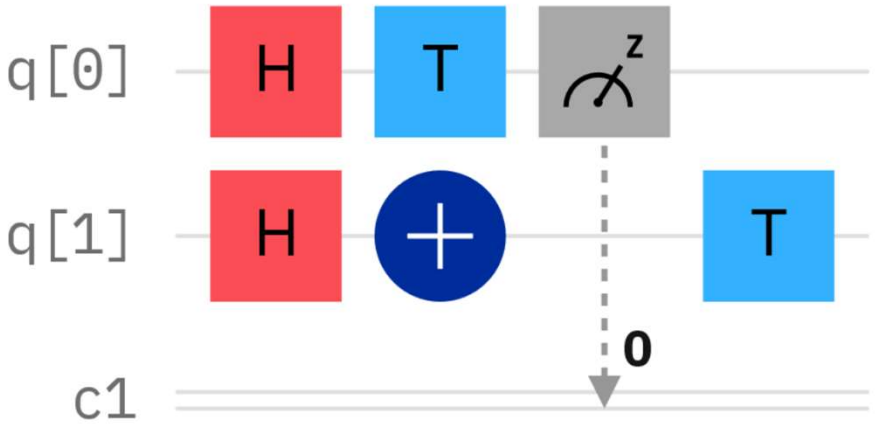
Ejemplo de visualización de transición de fase con la puerta general de cambio de fase

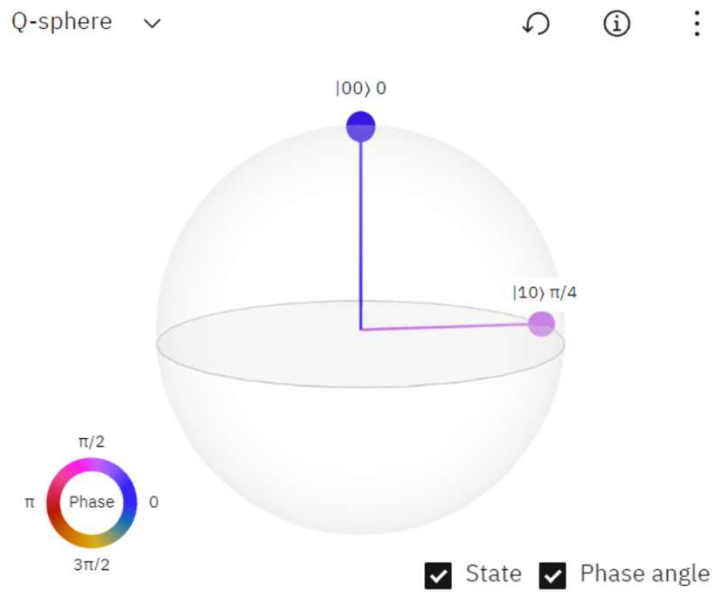


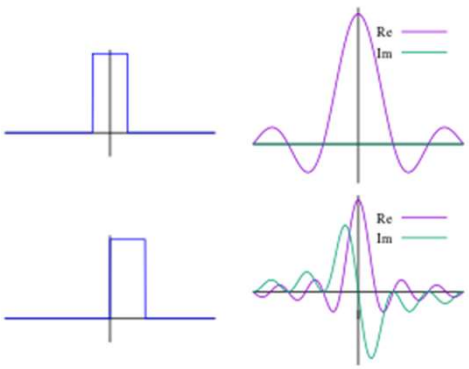


Transición de Fase en IBM Composer

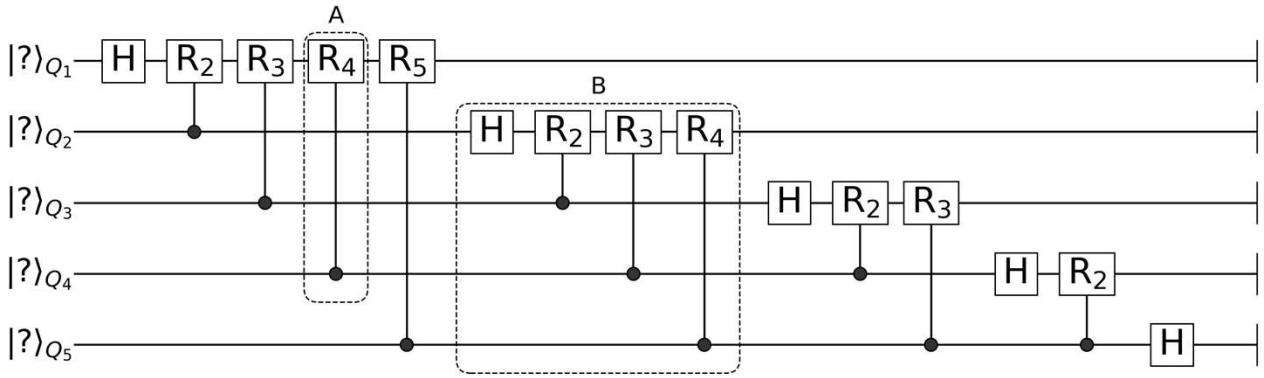
Ejemplo de visualización de transición de fase con la puerta general de cambio de fase







$$F(x) = \int_{-\infty}^{\infty} f(\xi) e^{2\pi i k x} d\xi$$



Quantum Fourier Transform (QFT)

QFT: Definición

- La Transformada Cuántica de Fourier (QFT) es una primitiva que nos permite acceder a los patrones ocultos y a la información almacenada dentro de la QPU.
- Como son **las fases y magnitudes relativas de un registro y hacerla legible**.
- La primitiva QFT tiene su propia forma de manipular las fases.
- Además de realizar la manipulación de la fase, también veremos que la primitiva QFT puede ayudarnos a calcular en superposición.

QFT: Definición

- La QFT es una Transformada Discreta de Fourier (DFT) en un ordenador cuántico.
- La DFT nos **permite inspeccionar las diferentes frecuencias contenidas dentro de una señal.**

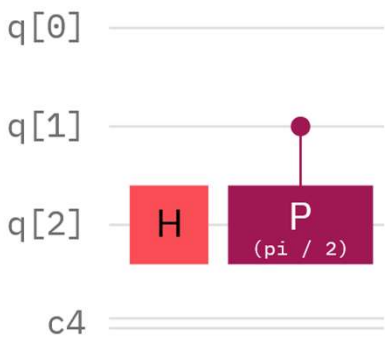
  La transformación DFT es básicamente idéntica a la mecánica matemática de la QFT.
- Una implementación rápida de la DFT muy utilizada es la transformada rápida de Fourier (FFT)

  Realiza la misma transformación que la DFT, solo que **opera sobre señales codificadas en registros cuánticos.**

QFT: Implementación 3 qubits

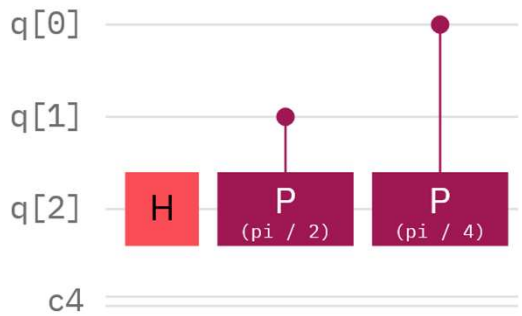
Vamos a implementar la Transformada Cuántica de Fourier (QFT) puerta a puerta

1. Este circuito puede implementarse utilizando puertas Hadamard en cada qubit, una serie de puertas P controladas, y puertas SWAP.



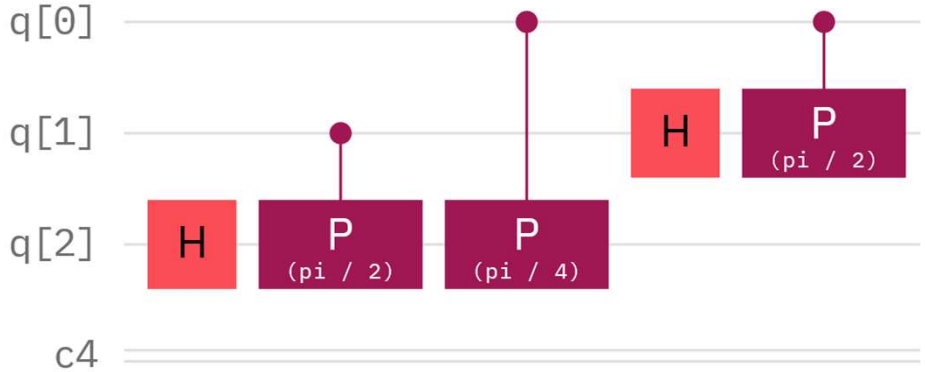
QFT: Implementación 3 qubits

2. Al igual que las puertas de un solo qubit, esta transformación que se aplica sobre un número determinado de qubits, es reversible aplicando las mismas puertas en orden inverso.



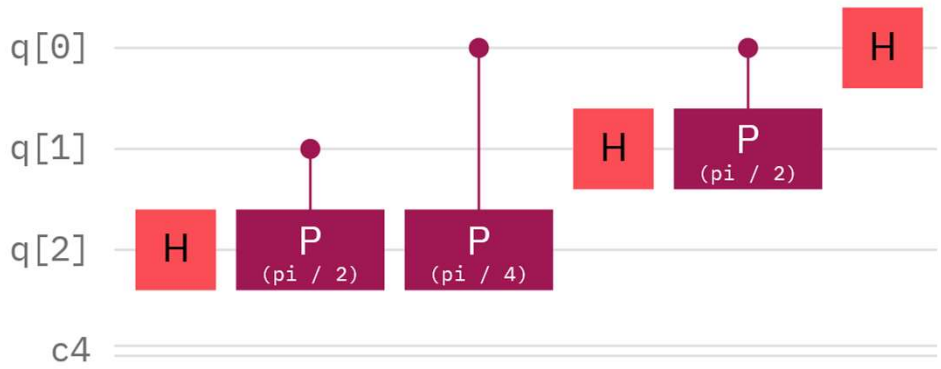
QFT: Implementación 3 qubits

2. Al igual que las puertas de un solo qubit, esta transformación que se aplica sobre un número determinado de qubits, es reversible aplicando las mismas puertas en orden inverso.



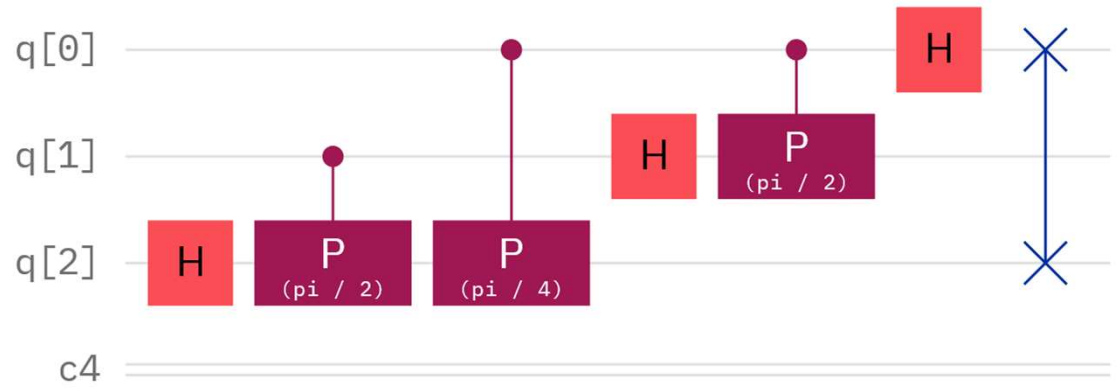
QFT: Implementación 3 qubits

3. Al igual que las puertas de un solo qubit, esta transformación que se aplica sobre un número determinado de qubits, es reversible aplicando las mismas puertas en orden inverso.



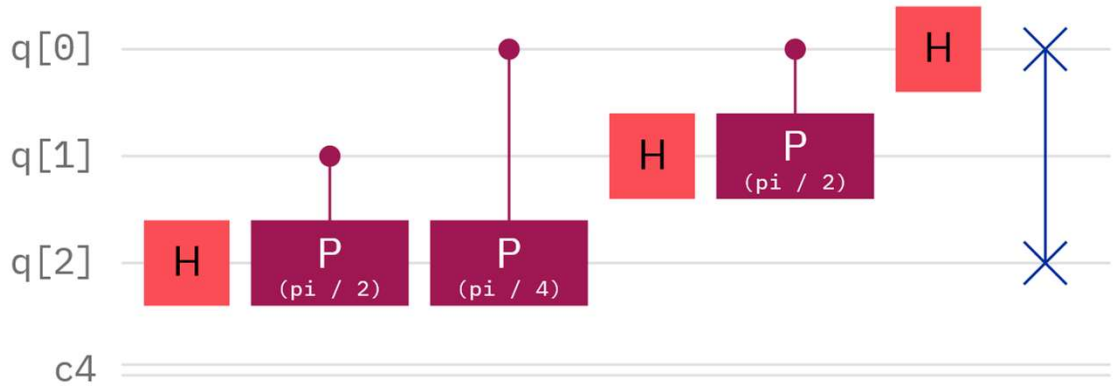
QFT: Implementación 3 qubits

3. Al igual que las puertas de un solo qubit, esta transformación que se aplica sobre un número determinado de qubits, es reversible aplicando las mismas puertas en orden inverso.



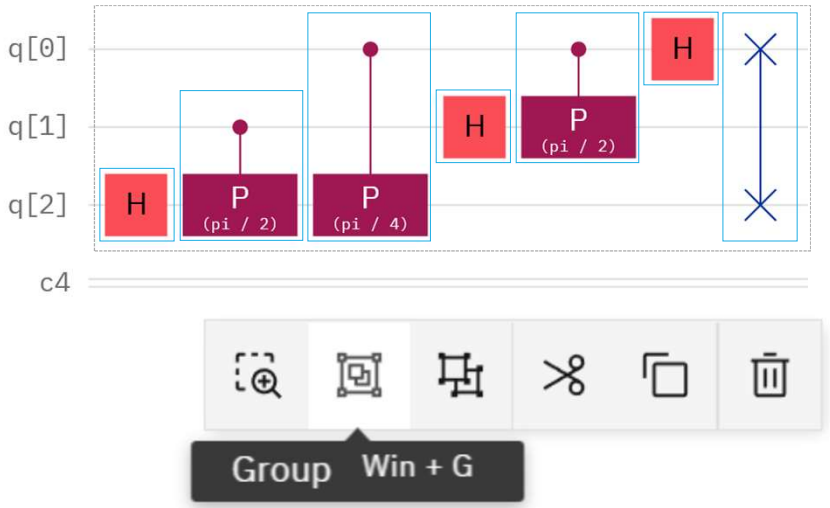
QFT: Implementación 3 qubits

Generalmente, la primitiva QFT suele verse representada en los circuitos **de forma agrupada como una única puerta** que empaqueta las puertas que componen la primitiva.



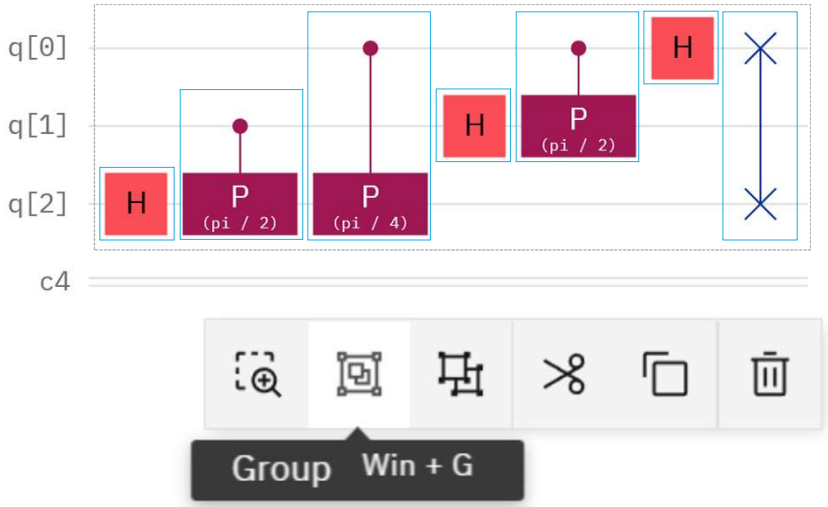
QFT: Implementación 3 qubits

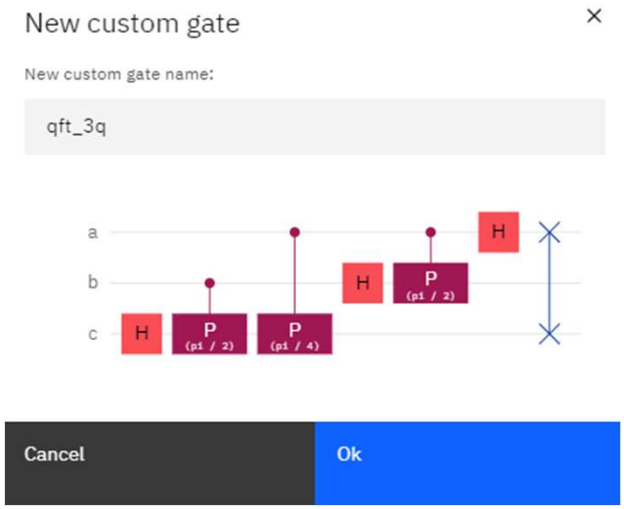
Generalmente, la primitiva QFT suele verse representada en los circuitos **de forma agrupada como una única puerta** que empaqueta las puertas que componen la primitiva.

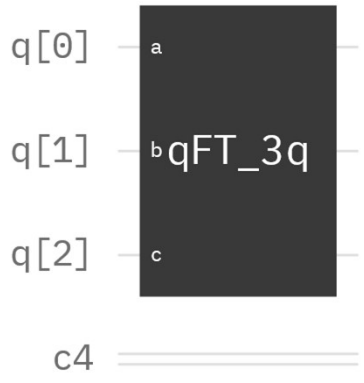


QFT: Implementación 3 qubits

Generalmente, la primitiva QFT suele verse representada en los circuitos **de forma agrupada como una única puerta** que empaqueta las puertas que componen la primitiva.

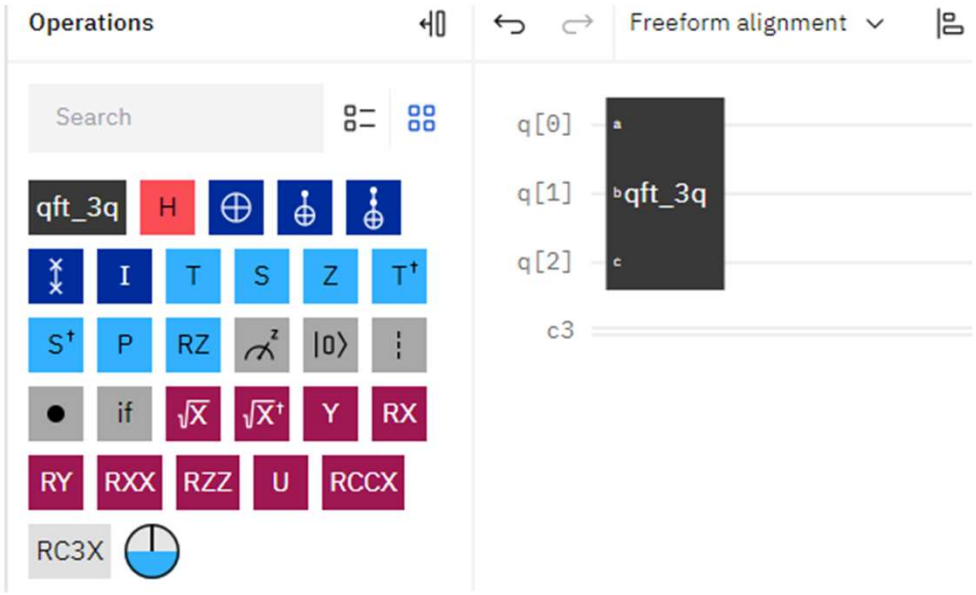






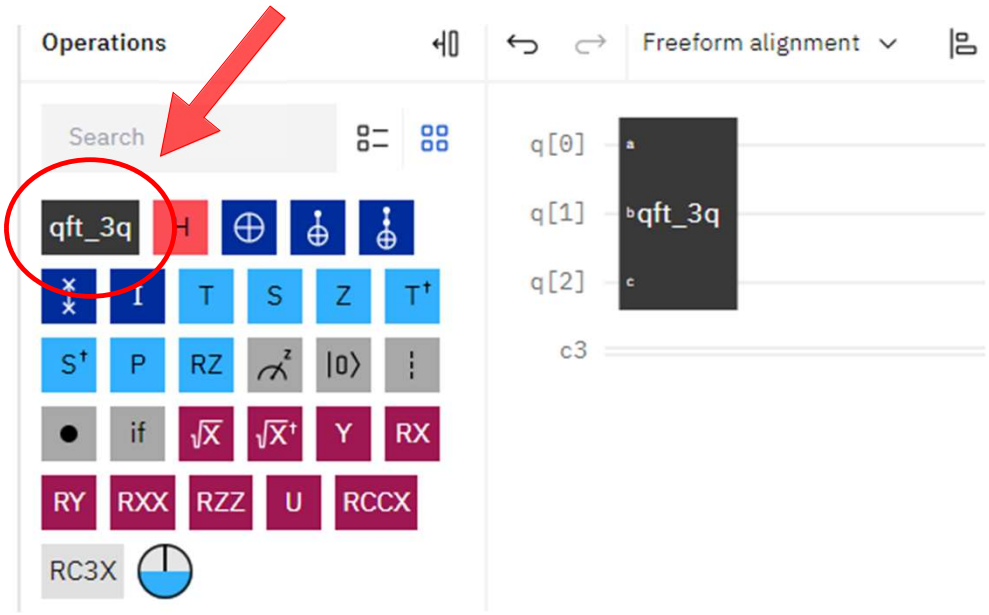
QFT: Implementación 3 qubits

Generalmente, la primitiva QFT suele verse representada en los circuitos **de forma agrupada como una única puerta** que empaqueta las puertas que componen la primitiva.



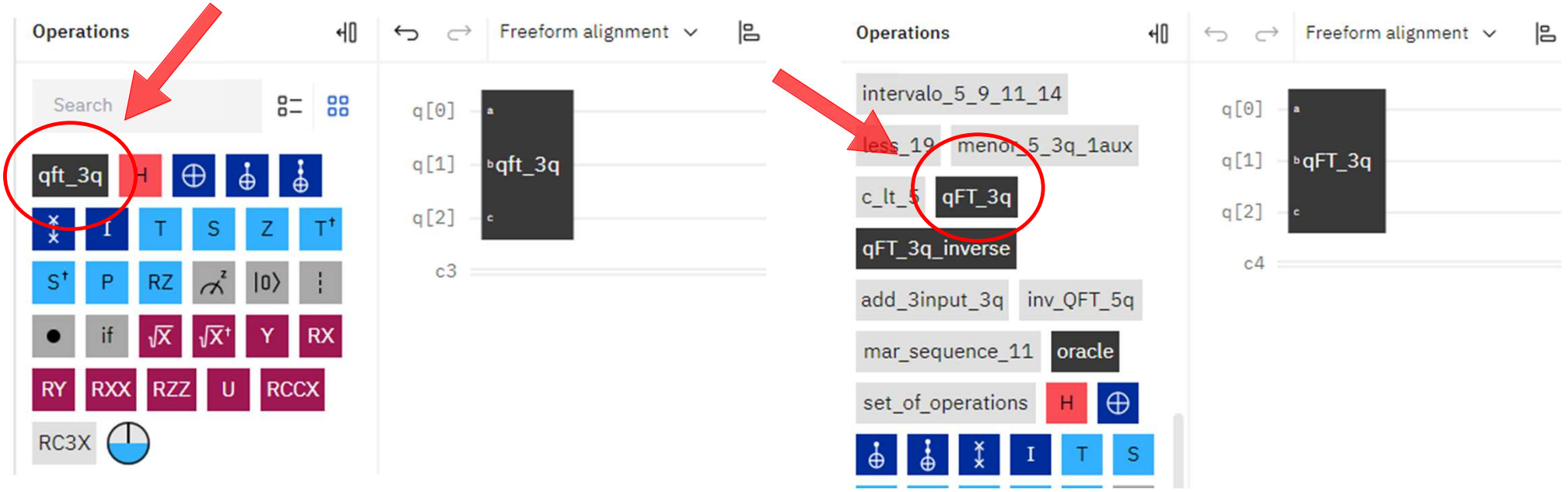
QFT: Implementación 3 qubits

Generalmente, la primitiva QFT suele verse representada en los circuitos **de forma agrupada como una única puerta** que empaqueta las puertas que componen la primitiva.

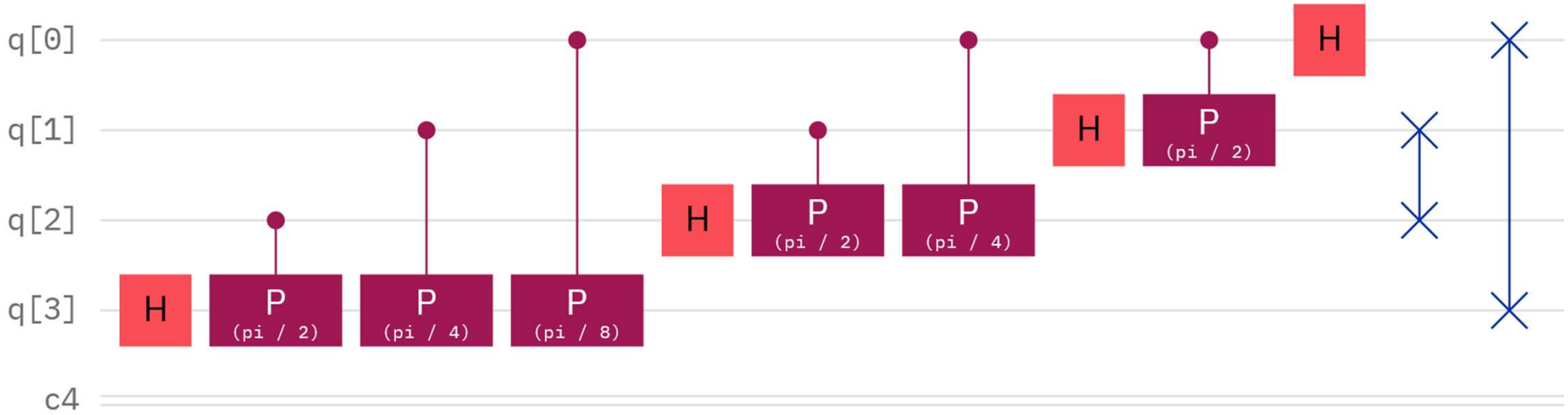


QFT: Implementación 3 qubits

Generalmente, la primitiva QFT suele verse representada en los circuitos **de forma agrupada como una única puerta** que empaqueta las puertas que componen la primitiva.



QFT: Implementación 4 qubits



THANK YOU
VERY MUCH FOR
YOUR
ATTENTION