192.036	VU Introduction to Quantum Computing	SS 2024
	Programming Exercise 1	Uwe Egly

1 General remarks

In all programming exercises, we use IBM's Qiskit system and the Python programming language to construct the quantum circuits, to simulate the quantum machine and to retrieve the results. Example programs are presented in the lecture's `tuwel` course.

2 General description

The goal is to construct a gate performing integer division. This `div` gate performs, for all $n > 0$ and all $m \geq 0$, the division of a non-negative $n + m$ qubit integer a and a positive n qubit integer b and returns the $n + m$ qubit quotient q and the n qubit remainder r such that $a = qb + r$ with $0 \leq r < b$. We annotate each register with its length given as a superscript. The `div` gate consists of $3n + m$ qubits $d_{3n+m-1} \cdots d_0$. The inputs are $a^{n+m} = d_{n+m-1} \cdots d_1 d_0$, $c^n = d_{2n+m-1} \cdots d_{n+m+1} d_{n+m} = 0 \cdots 0$ and $b^n = d_{3n+m-1} \cdots d_{2n+m+1} d_{2n+m}$. The `div` gate performs the action

$$\boxed{\left|b^n\right\rangle \left|c^n\right\rangle \left|a^{n+m}\right\rangle \xmapsto{\texttt{div}} \left|b^n\right\rangle \left|q^{n+m}\right\rangle \left|r^n\right\rangle .}$$

In order to implement the `div` gate, we provide the implementation of a subtraction (`subo`) gate with negation qubit and a controlled addition (`cadd`) gate.

A `subo` gate `subo_g` can be constructed by

```
subo_g = subo_gate(qAStart, qASize, qBStart, qBSize, neg).
```

A `cadd` gate `cadd_g` can be constructed by

```
cadd_g = cadd_gate(qAStart, qASize, qBStart, qBSize, ctrl).
```

For register `X`, the argument `qXStart` indicates its start position, `qXSize` its size and `neg`/`ctrl` indicates the position of the negation qubit and control qubit, respectively. Each call returns a gate which can be appended to a given circuit.

We recommend to use the most significant qubit named o for negation/control. The inputs for the gates are then $a = d_{n-1} \cdots d_1 d_0$, $b = d_{2n-1} \cdots d_{n+1} d_n$, and $o = d_{2n}$. The value o changes in the `subo` gate from 0 to 1 if $a - b < 0$. The actions of these gates are as follows.

$$|0\rangle\,|b\rangle\,|a\rangle \xmapsto{\texttt{subo}} |o\rangle\,|b\rangle\,|a-b\rangle$$

$$|0\rangle\,|b\rangle\,|a\rangle \xmapsto{\texttt{cadd}} |0\rangle\,|b\rangle\,|a\rangle \qquad |1\rangle\,|b\rangle\,|a\rangle \xmapsto{\texttt{cadd}} |1\rangle\,|b\rangle\,|a+b\rangle$$

3 Your tasks

We guide you step by step through the implementation process.

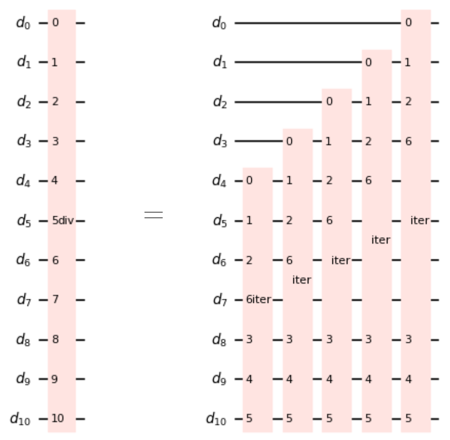
1. Construct a gate `iter` which will be used later on. This gate first subtracts with negation qubit the n qubit non-negative integer b from the n qubit non-negative integer a using the subtraction gate `subo`. In case the negation qubit o is 1, the difference is negative and we add b back. This is performed by a controlled addition with gate `cadd`, using the negation qubit o from the `subo` gate as the control qubit of the `cadd` gate. Finally the negation/control qubit is negated.

   The inputs of the `iter` gate are $a = d_{n-1} \cdots d_1 d_0$, $b = d_{2n-1} \cdots d_{n+1} d_n$, and negation/control qubit $o = d_{2n} = 0$. It performs the action

$(a - b) \geq 0$	$(a - b) < 0$
$\lvert 0\rangle \lvert b\rangle \lvert a\rangle \xmapsto{\texttt{iter}} \lvert 1\rangle \lvert b\rangle \lvert a-b\rangle$	$\lvert 0\rangle \lvert b\rangle \lvert a\rangle \xmapsto{\texttt{iter}} \lvert 0\rangle \lvert b\rangle \lvert a\rangle$

   Implement the gate for later use. Draw the gate for $n = 3$, test your implementation for all possibilities over $n = 2$ qubits and document the test cases.

2. Construct the `div` gate using the `iter` gate.

   The example circuit for $n = 3$ and $m = 2$ is given on the left. Let $a^{n+m} = d_4 d_3 d_2 d_1 d_0 = 11101$ and let $b^n = d_{10} d_9 d_8 = 111$. Figure out the input and output of each `iter` gate and document the findings in a table. What is q and r for this example? Test your implementation for various n and m and document the test cases.

3. Suppose we want to invert an n qubit number b. How can we do this using the `div` gate? How does m influence the precision of the inversion process? Invert $b^4 = 1011$ and discuss the result.