# A primer on quantum RAM

Olivia Di Matteo

June 19, 2020

**Abstract**

**This document is a work in progress** and is being continuously edited.

## Contents



# 1 Introduction and FAQ

Quantum RAM (qRAM) has gained some notoriety in the past few years, and attitudes toward it depend quite heavily on the circles you run in. The goal of this set of notes is to provide a general overview of the subject, as well as discuss some very specific implementations. It is meant to be a companion to the Q# qRAM libraries we are writing (cite the repo), but can also serve as a standalone reference.

In my experience, explaining qRAM to someone boils down to answering a handful of key questions. I'll provide some potentially unsatisfactory answers up front, but go into far more detail in the next few sections[1].

1. **Do I need a qRAM?** *Sometimes.* You'll need a qRAM, or some more general means of *quantum state preparation* in quantum machine learning (QML) algorithms that require you to load in classical data, or query an oracle that returns classical data. I've heard a number of stories of people working on QML being actively discouraged from doing so because "QML won't work without a qRAM". That's just not true, because *many QML algorithms do not need a qRAM.* Now, whether or not they yield any quantum advantage is a separate question, and won't be discussed here. The key point I want to make is that *some* QML algorithms need a qRAM, and they will potentially run into trouble as per the next question.

[1] Those details may not be satisfactory either.

2. **Can we design an efficient qRAM?** *Maybe.* In this primer we'll take a look at proposals that will in principle run in polynomial depth, and others that scale far worse. There are some very interesting qubit-time tradeoffs one can explore, in particular if the data being stored has some sort of underlying structure. Regardless, even if we can design an efficient circuit, we'd also like something that is efficient in a fault-tolerant setting, and this is potentially very expensive.
3. **Can I build one?** *Maybe.* No one has actually done so, but there are a handful of hardware proposals that will be discussed in more detail in section 6.

# 2 A brief history of the brief history of quantum machine learning

# 3 Encoding classical data on a quantum computer

There is a nice overview of the encodings below in [1].

## 3.1 Basis encoding

Give overview of the basis encoding, i.e.

$$\text{data } 01001 \rightarrow |01001\rangle \tag{1}$$

## 3.2 Quantum RAM / ROM

Query as bits in superposition:

$$\sum_i |i\rangle |0\rangle \rightarrow \sum_i |i\rangle |b_i\rangle \tag{2}$$

Query as phase in superposition (applications to Grover)

$$\sum_i |i\rangle \rightarrow \sum_i (-1)^{b_i} |i\rangle \tag{3}$$

## 3.3 Quantum state preparation (amplitude encoding)

Given some vector $\mathbf{a} = (a_0, \cdots, a_{n-1})$, create the quantum state

$$|\psi\rangle = \sum_{i=0}^{n-1} a_i |i\rangle \tag{4}$$

Discuss how this relies on having a qROM as an underlying subroutine.

# 4 Bucket-brigade quantum RAM

**[These paragraphs are lifted from my thesis and need to be edited and expanded on].**

Architectures for qRAM began to emerge roughly a decade ago with the bucket brigade model of [2, 3]. In this model, $2^n$ bits of classical data (that we would like to quantumly query) are stored in the $2^n$ leaves of a binary tree, as depicted in. The nodes of the tree relate to the address, with the $j$-th address bit corresponding to the $j$-th layer of the tree, as will be described below.

At each node is a three-level state (a qutrit), with levels denoted by $|\text{wait}\rangle, |\text{left}\rangle, |\text{right}\rangle$. All the qutrits begin in $|\text{wait}\rangle$. Address qubits are then sent through one by one and the $j$-th bit follows a path to the $j$-th layer. When a qubit reaches a qutrit in its destination layer, it changes the state of that qutrit. An address bit in 0 initiates a unitary operation that sends $|\text{wait}\rangle \rightarrow |\text{left}\rangle$, while a 1 sends $|\text{wait}\rangle \rightarrow |\text{right}\rangle$. To reach their destination layers, the qubit simply follows the path directed by the qutrits. After all the address qubits are sent, a quantum 'bus' traverses the path, couples to the desired memory cell to gather the data, and is reflected back the way it came. All the nodes are then re-initialized to the $|\text{wait}\rangle$ state.

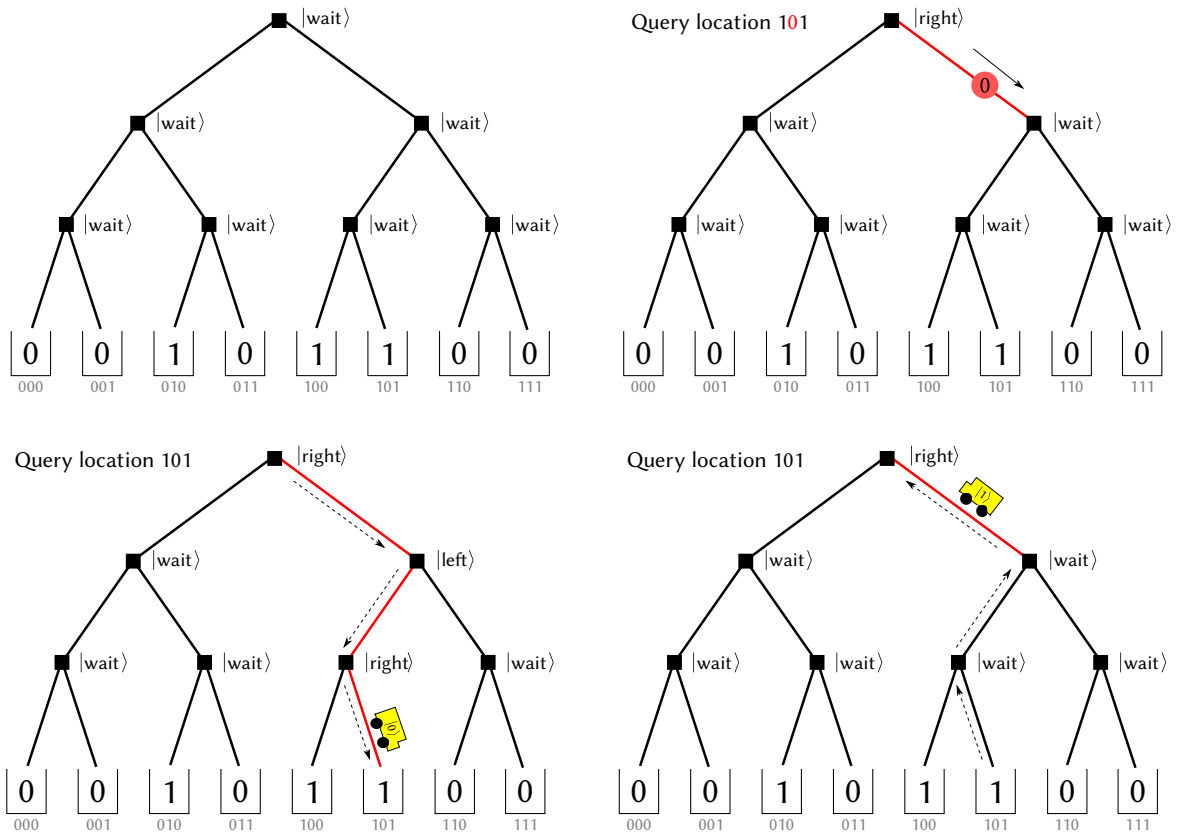


Figure 1: Schematic of a bucket brigade quantum RAM query. (Top left) Memory contents are stored in the leaves of a binary tree, whose nodes are denoted by qutrits with possible states $|\text{wait}\rangle$, $|\text{left}\rangle$, and $|\text{right}\rangle$. (Top right) A bucket brigade RAM query is set up by sequentially sending in address qubits to change the qutrit states until a path to the desired cell is created. (Bottom left) To extract the contents of the memory cell, a 'bus' photon is used. The photon is directed to the desired cell by the qutrits, where it couples to the memory cell and copies its contents. (Bottom right) The bus photon travels back through the qRAM, resetting the qutrit states to $|\text{wait}\rangle$ on its way out.

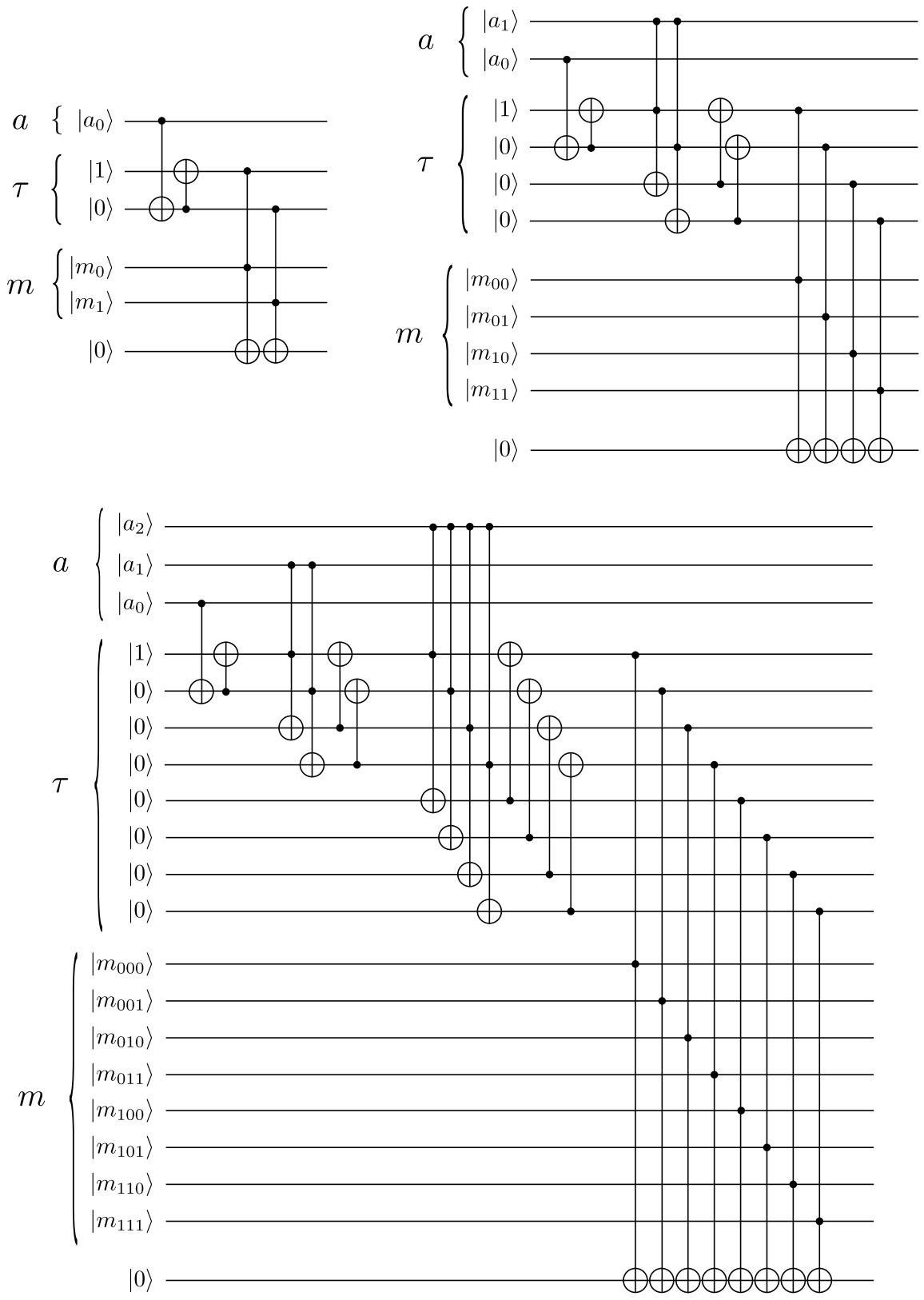


Figure 2: **For pedagogical purposes; not most optimal**. (Top left) A bucket brigade circuit for a 1-qubit address (2 memory cells). (Top right) A bucket brigade circuit for a 2-qubit address. The first 6 gates are a fanout of the address bits to the auxiliary 'trigger' register, denoted by $\tau$. A cascade of Toffolis then couples the memory cells to the output. To make the query fully reversible, the fanout component must be repeated. The subscripts on the memory cells are ordered from top to bottom of the address register, i.e. $a_{n-1}, a_{n-2}, \ldots, a_0$. (Bottom) A bucket brigade circuit for a 3-qubit address, highlighting the pattern that emerges in the construction of this circuit. The fanout component is recursive, with the 1- and 2-qubit fanout portions appearing.

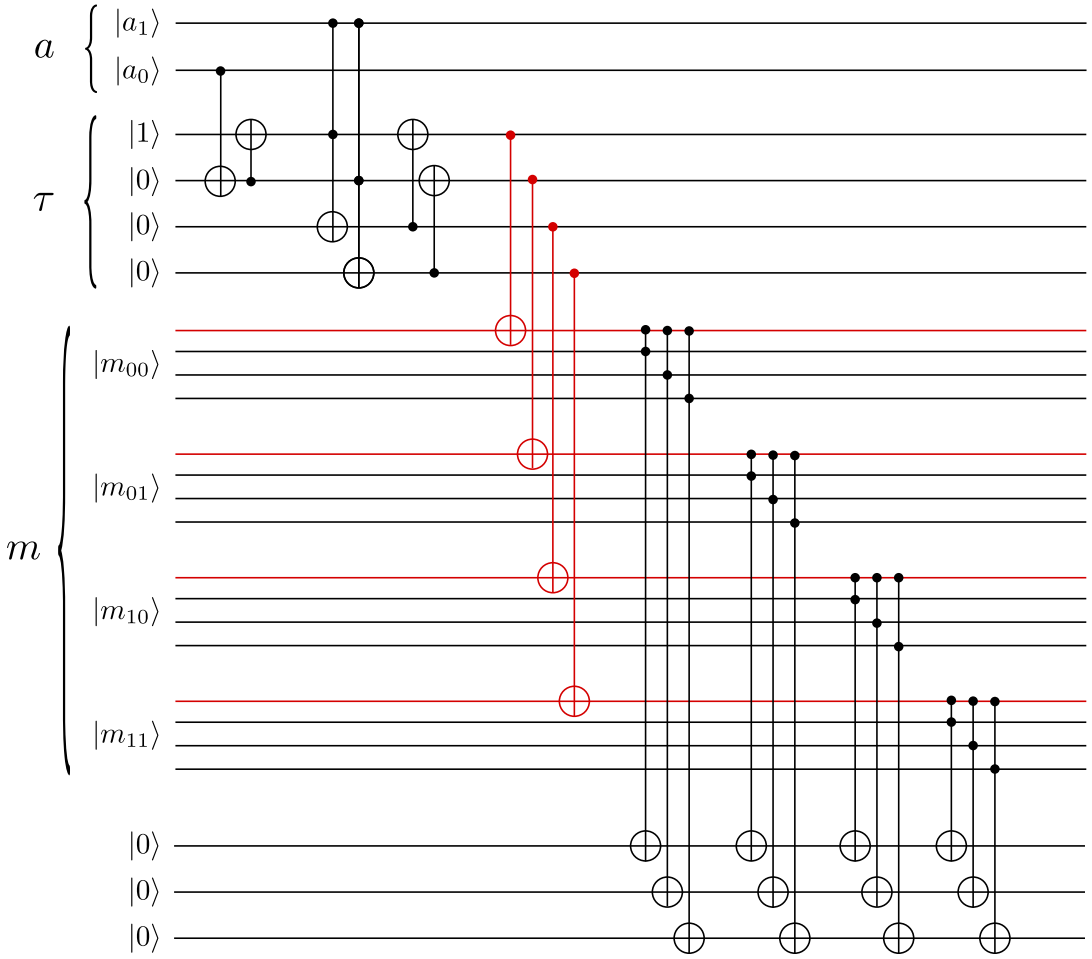


Figure 3: **For pedagogical purposes; not most optimal.** Extension of the bucket brigade circuit model to the case where each memory cell contains multiple output bits (in this case, 3). The structure of the circuit is largely the same, except the original trigger register acts on a set of 'indicator' qubits representative of each memory cell (highlighted in red). These indicator qubits then control coupling of the memory contents to the set of output bits.

### 4.1 How it works

### 4.2 Bit query circuit model

### 4.3 Phase query circuit model

### 4.4 Analysis

## 5 Quantum ROM

### 5.1 Quantum ROM

### 5.2 Efficient state preparation with a qROM

Discuss [4]

### 5.3 Optimizing quantum ROMs

### 5.4 Problem-specific qROMs

Talk about the qROM in [5]

## 6 Hardware proposals for qRAM

## 7 Outlook

## References


[1] Maria Schuld and Francesco Petruccione. *Supervised Learning with Quantum Computers*. Quantum Science and Technology. Springer International Publishing, Cham, apr 2018.

[2] Vittorio Giovannetti, Seth Lloyd, and Lorenzo Maccone. Quantum random access memory. *Phys. Rev. Lett.*, 100:160501, Apr 2008.

[3] Vittorio Giovannetti, Seth Lloyd, and Lorenzo Maccone. Architectures for a quantum random access memory. *Phys. Rev. A*, 78:052310, Nov 2008.

[4] Guang Hao Low, Vadym Kliuchnikov, and Luke Schaeffer. Trading T-gates for dirty qubits in state preparation and unitary synthesis. *arXiv e-prints*, page arXiv:1812.00954, Dec 2018.

[5] Ryan Babbush, Craig Gidney, Dominic W. Berry, Nathan Wiebe, Jarrod McClean, Alexandru Paler, Austin Fowler, and Hartmut Neven. Encoding Electronic Spectra in Quantum Circuits with Linear T Complexity. *arXiv:1805.03662 [cond-mat, physics:physics, physics:quant-ph]*, May 2018. arXiv: 1805.03662.