QMPROM: Quantum Technology for Multivalued Qubit Storage using Programmable Read Only Memory

Tamanna Tabassum
Ahsanullah University of Science and Technology,
141 & 142, Love Road, Dhaka-1208, Bangladesh

Fatema Akter
Green University of Bangladesh, Purbachal,
American City, Kanchon 1460, Dhaka, Bangladesh

ABSTRACT
Compared to traditional multivalued computing systems, multivalued quantum computing offers more processing power and uniqueness thanks to its combination of theoretical computer science and quantum physics. It uses quantum mechanics and the collective features of quantum states, such as superposition with entanglement, and interference, to accomplish some calculations with unprecedented computational speed. When dealing with these intricate issues, quantum algorithms adopt a novel strategy: they generate multidimensional spaces from which the patterns connecting distinct data points arise. These algorithms could effectively address complex mathematical problems, generate secure codes that are difficult to crack, and predict interactions among multiple particles in chemical reactions. They also refer to memory devices, a general term for integrated circuits that can be programmed in a lab to carry out intricate functions. Quantum computers outperform traditional Turing machines by coherent superposition of states. Large photonic quantum processing systems would be possible with the help of quantum memories since they would be capable of effectively modifying, buffering, and adjusting the timing of photonic signals. While qubits can only be used as input states in certain types of computers, quantum computing allows for the development of new computer types that have higher storage capacities despite the slower performance of regular programmable read-only memory (PROM). The design of multivalued quantum-based PROM is a key concern in order to produce affordable, durable, high-capacity, reliable, and energy-efficient memory systems. This study presents the construction of a multivalued PROM architecture based on quantum mechanics, utilizing algorithms for multiple valued quantum ternary operations.

Keywords
Quantum Logic; Ternary; Parallel Processing; Multivalued; Qutrit; QMPROM.

1. INTRODUCTION
Quantum computing focuses on developing computer technology based on the principles of quantum theory, which explains the behavior of matter and energy at atomic and subatomic levels [1]. It is well known that there are aspects of quantum mechanics that are absent from classical physics. These peculiar characteristics include decoherence, which asserts that when a coherent (superposed) state interacts with its surroundings, it transforms into a classical state devoid of superposition, and superposition, which is the ability to exist in numerous states [2, 3]. Hence, entanglement: the state in which two or more particles can be connected and, if so, alter one another's properties must be fully isolated from the outside world for a quantum computer to operate with superposed states [4, 5]. In a quantum system, it is impossible to predict with any degree of precision every property of a particle. In theory, given enough time, any issue that a quantum computer can answer may also be solved by a classical computer [6]. When dealing with these kinds of intricate issues, quantum algorithms adopt a novel strategy: they create multidimensional spaces where the patterns connecting distinct data points appear [7]. Rather than using the more widely used binary system (base 2) for calculations, ternary computers use ternary logic, or base 3 [8].

Multivalued quantum computing, or ternary quantum computing, differs from other traditional computing systems due to its rapid processing speeds and parallel processing capabilities [9, 10]. This means that it employs trits rather than bits, in contrast to conventional memory, which stores information as ternary states (represented by "|2>"s, "|1>"s, and "|0>"s). The main goal is to create multivalued quantum-based programmable ROMs that offer low-cost, robust, high-density, dependable, and energy-efficient memory technologies. One of the earliest calculators was made completely of wood and ran on balanced ternary, according to Thomas Fowler's 1840 construction [11]. At Moscow State University in the Soviet Union, Nikolay Brusentsov created Setun, the first electronic ternary computer, in 1958. Compared to succeeding binary computers, Setun provided a number of advantages, including lower manufacture and energy costs. In 1970, Brusentsov improved the model and called it Setun-70 [12]. The binary machine-based ternary computer emulator Ternac was first released in the United States in 1973. Furthermore, another ternary computer, the QTC-1, was created in Canada [13]. Key elements of photonics-based quantum technologies include single-photon detectors, frequency converters, photon sources, quantum random number generators, and quantum memory [14, 15]. The ability to store and recover the quantum state of a single photon is the main subject of this essay on quantum memory. The diverse approaches to quantum memory cover a broad spectrum of electromagnetic interactions and present the most recent developments in quantum control of optical signals [16, 17]. Certain principles are evident when it comes to memory parameters: a greater cost per bit of stored information results from lowering access time, whereas a lower cost is associated with increased memory capacity [18]. A memory system's capacity, sometimes known as memory volume, is the total number of locations within it. Capacity can be measured in bits, bytes, or words; therefore, it is crucial to define the length of a word in terms of bits or bytes [19, 20]. The time elapsed between submitting a memory access request and getting the relevant data is known as memory access time. A memory unit's access time affects its speed, which is measured as the amount of time it takes to retrieve a single unit of data; quicker memory has shorter access times [21]. Memory cycle time is another term for the smallest amount of time needed between successive access requests to the same memory address. Memory transfer rate, expressed in bits per second or bytes per second, indicates how quickly data can be read from or written to the designated memory [22]. PROM and programmable logic are often categorized within the same circuit type. The architecture of Programmable Logic Devices (PLDs) provides greater flexibility compared to PROM architecture, which tends to reach its limitations when numerous inputs are connected to multiple outputs. Data or programs can be written only once, but once written, they can be read as often as needed [23, 24].

A PROM chip, which has non-volatile memory and can hold up to 4 megabytes (MB) of data per chip, is mostly employed during a modern computer's startup procedure [25]. Unrestricted access to any place and time across the entire address space is made possible by random memory. Regardless of the sequence of all prior accesses, the access is possible separately. An address may be accessed in any sequence. In a random-access memory, access is provided by separate hardware circuits at each place. Address decoding results in the activation of these circuits [26]. A programmable OR gate array, which may be conditionally reversed to generate an output, is connected to a static AND gate array in programmable read-only memory. While the idea of a PROM and ROM are similar, a PROM does not create all of the minterms and does not offer complete variable decoding [27, 28]. PROM devices use arrays of transistor cells configured in a "fixed-OR, programmable-AND" fashion to generate "sum of products" binary logic equations for each output based on the inputs and either synchronous or asynchronous feedback from the outputs. To translate basic code into the commands required by a device programmer for design implementation, system designers can use development software. Developing memory technologies that are low-cost, resilient, high-density, dependable, and energy-efficient is a major problem in the development of multivalued quantum-based PROMs.

2. LITERATURE STUDY

In this section, the basics of quantum technology, multivalued basic gate operations with the algorithms, and quantum ternary-based storage PROM are discussed in detail.

2.1 Multivalued Quantum Computing

In ternary quantum computing, one qubit is employed as an output and two qubits as inputs. Qutrit states are the name given to these basis states, which are represented by 3 × 1 vectors [29]:

$$|0\rangle = \begin{bmatrix}1\\0\\0\end{bmatrix} \quad |1\rangle = \begin{bmatrix}0\\1\\0\end{bmatrix} \quad |2\rangle = \begin{bmatrix}0\\0\\1\end{bmatrix}$$

The following formula, which is the linear superposition of the previously mentioned base states, defines a qutrit in a ternary quantum system:

$$\psi = \alpha\,|0\rangle + \beta\,|1\rangle + \gamma\,|2\rangle$$

Here α, β, and γ are the complex values that represent the probability values of the basis states, and ψ is the wave function, and ψ is the wave function.

2.2 Quantum Ternary Gates

2.2.1 Quantum Ternary Shift gates

Commonly utilized are six triple permutation matrices, which are also known as quantum triple shift gates. Zero is the primary state. The numerals 0, 1, and 2, respectively, are in the columns. The qutrit levels shift by 1 when transformed by Z (+1). The qutrit states are shifted by two when Z (+2) is transformed. Transform Z (01) trades |0> and |1>, Transform Z (02) exchanges |0> and |2>, and Transform Z (12) swaps the qutrit values |1> and |2>.

$$Z_3(+0) = \begin{bmatrix}1&0&0\\0&1&0\\0&0&1\end{bmatrix} \quad Z_3(+1) = \begin{bmatrix}0&0&1\\1&0&0\\0&1&0\end{bmatrix} \quad Z_3(+2) = \begin{bmatrix}0&1&0\\0&0&1\\1&0&0\end{bmatrix}$$

$$Z_3(12) = \begin{bmatrix}1&0&0\\0&0&1\\0&1&0\end{bmatrix} \quad Z_3(01) = \begin{bmatrix}0&1&0\\1&0&0\\0&0&1\end{bmatrix} \quad Z_3(02) = \begin{bmatrix}0&0&1\\0&1&0\\1&0&0\end{bmatrix}$$

Fig 1: 1-qutrit ternary permutation transformations

These transformations are depicted in Figure 1, and the operations of 1-qutrit ternary shift gates are given in Table 1.

Table 1. 1-qutrit Ternary Shift Gates Operations

A	|0>	|1>	|2>
Z (0) = A	|0>	|1>	|2>
Z (+1) = A+1	|1>	|2>	|0>
Z (+2) = A+2	|2>	|0>	|1>
Z (12) = 2A	|0>	|2>	|1>
Z (01) = 2A+1	|1>	|0>	|2>
Z (02) = 2A+2	|2>	|1>	|0>

2.2.2 Quantum Ternary Toffoli gates

Another quantum ternary gate is the ternary Toffoli gate [30]. Its inputs are A, B, and C, with C serving as the controlled input, while A and B act as the controlling inputs. The outputs are P = A and Q = B. Figure 2 displays the symbol for the generalized 3-qutrit ternary permutation/shift operations.

$$R = \begin{cases} \text{Z transforms of C; if } A = X_1 \text{ and } B = X_2 \\ C \quad ; \quad \text{otherwise} \end{cases}$$

The ternary Toffoli gate has the following outputs: P, Q, and R. Inputs A and B correspond to X_1 and X_2, then outputs P and Q are equivalent to A and B, while output R represents the Z transform of C, where Z = {+1, +2, 01, 02, 12}. If this condition is not met, the input C and output R are the same. The Toffoli gate for a single regulated input is depicted in Figure 3. In this case, the gate will only open if bit 2 is the controlled bit.

2.2.3 Quantum Ternary C^2NOT gates

Multi-qutrit control operations are feasible in ternary logic. A revised definition of 3-qutrit C^2 NOT was provided by [31], and it is utilized to implement the ternary midterm simplification rules.

$$C^2NOT\,(A, B, C) = \begin{cases} NOT\,(C) \;;\; \text{if } A \,!=\, B \text{ and } A,B \,!=\, 0 \\ C \;;\; \text{otherwise} \end{cases}$$

Here the target input is C and the control inputs are A and B. The (+1) action of the supplied input will be provided in this case by the last NOT (C). Figure 4 displays the symbol for the ternary C^2 NOT gate.

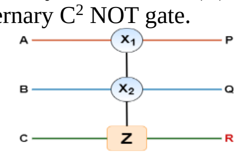

Fig 2: 3-qutrit ternary Permutation Operations

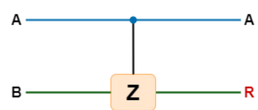

Fig 3: Ternary 1-bit Controlled Operation

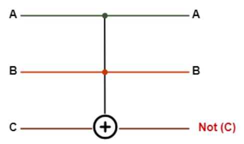

Fig 4: Ternary 3-qutrit C^2 NOT gate

2.3 Ternary Quantum Basic Logic Gates

Functions with ternary logic are ones whose relevance increases when a third value is familiar with the binary logic. Here, the ternary levels for fundamental logic gates are represented by the numbers 0, 1, and 2, which stand for false, undefined, and true, respectively. The following definitions [32, 33] apply to the fundamental operations of ternary logic:

$$y_{OR} = max(x, y)$$
$$y_{NOR} = \overline{max(x, y)}$$
$$y_{AND} = min(x, y)$$
$$y_{NAND} = \overline{min(x, y)}$$
$$y_{XOR} = sum(x, y)$$
$$y_{XNOR} = \overline{sum(x, y)}$$

Where |x>, |y>= {|0>, |1>, |2>}

The minimal value of the AND logic gate's inputs determines the value of the gate's output. Similar to this, the output value of an OR logic gate is determined by the highest input value.

2.3.1 Quantum Ternary OR Operations

Y_{OR} = max (X, Y) is the definition of the quantum OR operation, where inputs of X and Y represent from the set {|0>, |1>, |2>}. The shifting operations of (+1) and (+2) shifting are necessary for the Quantum Ternary OR operation. When two inputs govern two (+1) operations, while two additional (+1) operations remain uncontrolled. Every input controls two (+2) procedures. Additionally, in order to obtain the anticipated outcome that was previously displayed in Table 2, a two-input controlled C^2NOT gate is required.

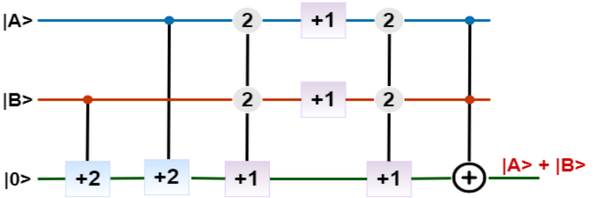

Figure 5: Ternary Quantum OR Gate

Figure 5 shows that the inputs B and A, respectively, control the first (+2) and second (+2) operations, which will only open if the input is |2>. Both A and B inputs control the following (+1), and they will only open if both inputs are |2>. There is no control over the next two (+1). Once more, two inputs govern the following (+1), which will only open if |2> is present in both inputs. And only in the event that A, B!= 0 && A!= B will the last XOR gate open.

Table 2. Quantum Ternary OR Operations

A	|0>	|0>	|0>	|1>	|1>	|1>	|2>	|2>	|2>
B	|0>	|1>	|2>	|0>	|1>	|2>	|0>	|1>	|2>
(1) +12	|0>	|0>	|2>	|0>	|0>	|2>	|0>	|0>	|2>
(2) +2	|0>	|0>	|2>	|0>	|0>	|2>	|2>	|2>	|1>
(1) +1	|0>	|0>	|2>	|0>	|0>	|2>	|2>	|2>	|0>
A +1	|1>	|1>	|1>	|2>	|2>	|2>	|0>	|0>	|0>
B +1	|1>	|2>	|0>	|1>	|2>	|0>	|1>	|2>	|0>
(1) +1 +1	|0>	|0>	|2>	|0>	|1>	|2>	|2>	|2>	|0>
C^2NOT	|0>	|1>	|2>	|1>	|1>	|2>	|2>	|2>	|0>

2.3.1.1 *Algorithm of Ternary Quantum OR Gate*

Algorithm 1: Multiple-Valued Quantum-Based OR Gate
Input: A, B; Output: |0>, |1>, |2>

```
Begin
Procedure DO_Quant_T-OR( | A >, | B >)
| M > <- perform_PlusTwoOp( | B >, | 0 > );
| N > <- perform_PlusTwoOp( | A >, | M > );
| O > <- perform_PlusOneOp2( | A >, | B >, | N > );
    | A1 > <- perform_PlusOneOp( | A > );
    | B1 > <- perform_PlusOneOp( | B > );
| P > <- perform_PlusOneOp2( | A1 >, | B1 >, | O >);
| Q > <- perform_C2NOTOp( | A1 >, | B1 >, | P > );
end procedure
Procedure perform_PlusTwoOp2( | A >, | B >, | C > )
    if the value of | A > and | B > both are | 2 >
        if the value of | C > is | 0 >
            return | 2 >
        else if the value of | C > is | 1 >
            return | 0 >
        else return | 1 >
        else return | C >
end procedure
Procedure perform_PlusTwoOp( | A >, | B >)
        if the value of | A > is| 2 >
        if the value of | B > is | 0 >
            return | 2 >
        else if the value of | B > is | 1 >
            return | 0 >
        else return | 1 >
        else return | B >
end procedure
Procedure perform_PlusOneOp2( | A >, | B >, | C > )
    if the value of | A > and | B > both are | 1 >
        if the value of | C > is | 0 >
            return | 1 >
        else if the value of | C > is | 1 >
            return | 2 >
        else return | 0 >
        else return | C >
end procedure
Procedure perform_C2NOTOp( | A >, | B >, | C > )
        if the value of |A> and |B> both are not
equal and not equal|0>
        if the value of | C > is | 0 >
            return | 1 >
        else if the value of | C > is | 1 >
            return | 2 >
        else return | 0 >
        else return | C >
end procedure
End
```

2.3.2 Quantum Ternary OR Operations

Y_{AND} = min (X, Y) is the definition of the quantum AND operation, where inputs of X and Y represent from the set {|0>, |1>, |2>}. Shifting operations such as (+1) and (+2) shifting are necessary for the Quantum Ternary AND operation. In this case, both inputs control one (+2) action, which will only open if the value of both inputs is |2>. The next step is to employ a C^2NOT gate that will only function if the A and B inputs satisfy the requirements A!=B and A, B!= 0. Both inputs control the next (+1) procedure, which will only open if the value of both inputs is |1>. As a result, the anticipated results are displayed in Table 3. This is the schematic diagram for the Quantum Ternary AND function is illustrated in Figure 6. Shifting operations such as (+1) and

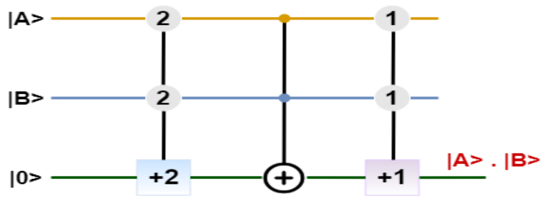

Figure 6: Ternary Quantum AND Gate

(+2) shifting are necessary for the Quantum Ternary AND operation. In this case, both inputs control one (+2) action, which will only open if the value of both inputs is |2>. The next step is to use a C^2NOT gate, the gate will operate only if the A and B inputs meet the conditions $A \neq B$ and $A, B \neq 0$.

Table 3. Quantum Ternary AND Operations

A	B	+2	C^2 NOT	Output (+1)
|0>	|0>	|0>	|0>	|0>
|0>	|1>	|0>	|0>	|0>
|0>	|2>	|0>	|0>	|0>
|1>	|0>	|0>	|0>	|0>
|1>	|1>	|0>	|0>	|1>
|1>	|2>	|0>	|1>	|1>
|2>	|0>	|0>	|0>	|0>
|2>	|1>	|0>	|1>	|1>
|2>	|2>	|2>	|2>	|2>

2.3.2.1 Algorithm of Ternary Quantum AND Gate

Algorithm 2: Multiple-Valued Quantum-Based AND Gate
Input: A, B ; Output: |0>, |1>, |2>

```
Begin
Procedure DO_Quant_T-AND( | A >, | B >)
 | M > <- perform_PlusTwoOp2( | A >, | B >, | 0 > );
 | N > <- perform_C2NOTOp( | A >, | B >, | M > );
 | Q > <- perform_PlusOneOp1( | A >, | B >, | N > );
 end procedure
Procedure perform_PlusTwoOp2( | A >, | B >, | C > )
    if the value of | A > and | B > both are | 2 >
            if the value of | C > is | 0 >
                    return | 2 >
            else if the value of | C > is | 1 >
                    return | 0 >
            else return | 1 >
            else return | C >
 end procedure
Procedure perform_PlusOneOp1( | A >, | B >, | C > )
    if the value of | A > and | B > both are | 1 >
            if the value of | C > is | 0 >
                    return | 1 >
            else if the value of | C > is | 1 >
                    return | 2 >
            else return | 0 >
            else return | C >
 end procedure
End
```

3. MULTIVALUED PROGRAMMABLE READ ONLY MEMORY (MPROM)

The parallel execution along with the rapid processing capabilities of multivalued quantum computing set it apart from other traditional systems for computation. Multivalued Quantum Programmable Read-Only Memory is shortened to MQPROM. It describes memory chips that integrate multivalued quantum OR and multivalued quantum decoder functionalities onto a single integrated circuit (IC) to store permanent or semi-permanent data [34, 35]. Multivalued quantum PROM's components are non-volatile; they continue to exist even after the computer is powered down. Figure 7 illustrates a block schematic of a multivalued PROM. There are k lines for input and n lines for output in it. The multivalued PROM is initially a combinational circuit that has multiple ternary OR gates equivalent to the unit's outputs and multiple ternary AND gates interconnected for the multivalued decoder.

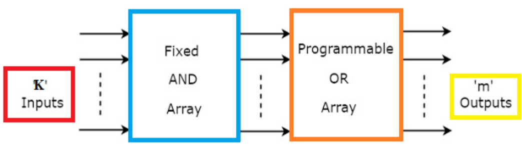

Figure 7: 3^k-to-m PROM Block

The output functions in PROM (n output lines with k input lines) will be computed by adding the minterms. With k input variables, 3^k several addresses are possible. Since a multivalued PROM has 3^k distinct addresses, each of which can be used to select an output word, the unit is said to store 3^k distinct words. The address value supplied to input determines a word accessible at the output lines at any given time. Consequently, the number of words (3^k) and bits per word (n) that make up a ternary PROM. The PROM circuit will be referred to as a 9-to-2 multivalued quantum PROM with output (n) = 2 and input (k) = 2. The function outputs, |F1> and |F2>, are respectively in the sum of minterms form, ∑ (0, 1, 2, 3) and ∑ (4, 5, 6, 7, 8).

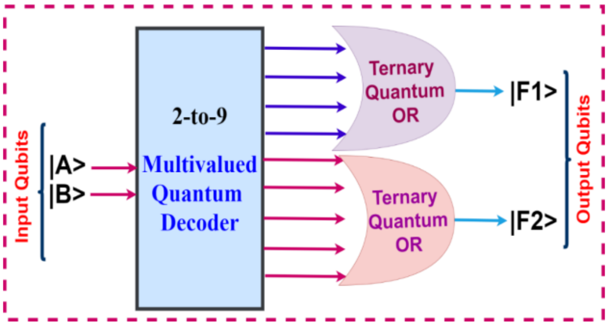

Figure 8: General Form of the Circuit Diagram of 9:2 QMPROM

Consider the general block diagram (Figure 8) for a multivalued quantum 9:2 PROM. Nine words total, divided into two input sequences (|A>, |B> = |0> or |1> or |2>), make up the unit. This suggests that the unit stores nine different word sequences, each of which may be transmitted to one of the two output lines (|F1> and |F2>).

3.1 Fundamental Component Structure of QMPROM

3.1.1 Multivalued Quantum 2:9 Decoder:

A ternary decoder is a combinational circuit with "n" lines of inputs and up to 3^n lines of output. When the decoder is enabled, among these outputs, active high will be one depending on the combination of inputs present. When the decoder is activated, as shown in Table 4, its outputs are just the minterms of 'n' input variable lines. Figure 9 depicts the block structure of a 1:3 ternary decoder, which accepts an input A and generates three outputs D_0, D_1 and D_2.

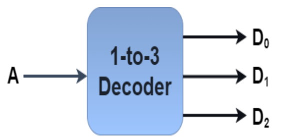

Figure 9: Block diagram of ternary 1:3 decoder

Table 4: Ternary 1:3 Decoder

A	D_0	D_1	D_2
0	2	0	0
1	0	2	0
2	0	0	2

The shifting operations needed for the Quantum Ternary 1:3 decoder operation consist solely of (+2) shifting. In this scenario, three (+2) operations are input-controlled and three (+2) operations are not. Figure 10 illustrates the circuit diagram for the Quantum Ternary 1:3 decoder operation.

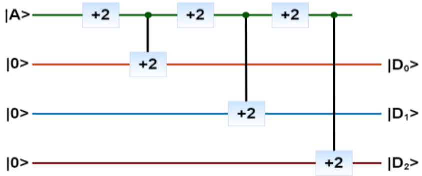

Figure 10: Quantum ternary 1:3 Decoder

From Figure 10, the 2nd (+2), 4th (+2), and 6th (+2) operations are controlled by the input that originally came from input A, followed by three uncontrolled (+2) operations, which will open only if the input is |2>. For A = 0, the 1st (+2) will produce an output of 2, which will open the 1st controlled (+2) and will provide the output of 2. Thus, $|D_0>$ will open. Others will remain closed as they will not produce the output of 2. For A = 1, the 1st (+2) will produce an output of 0, and the 2nd uncontrolled (+2) will produce an output of 2, which will open the 2nd controlled (+2) and will provide the output

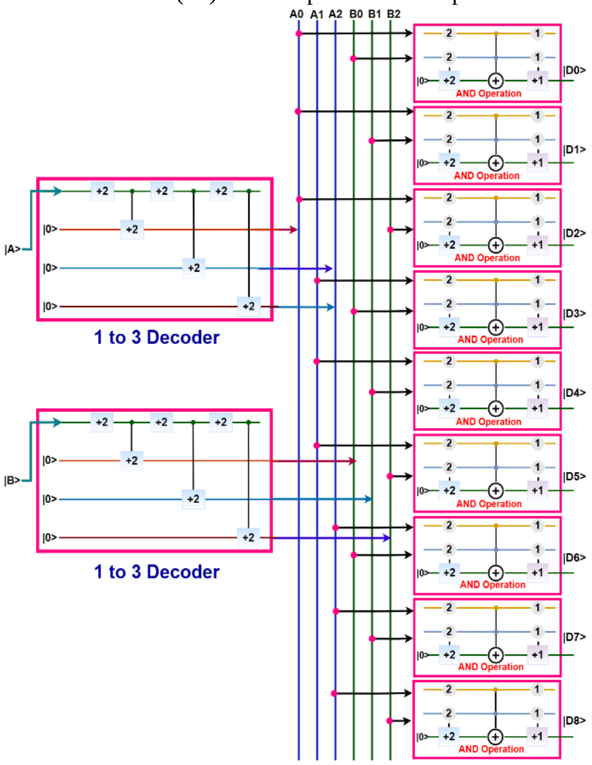

Figure 11: Quantum ternary 2:9 Decoder

of 2. Thus, $|D_1>$ will open. Others will remain closed as they will not produce the output of 2. For A = 2, the 1st uncontrolled (+2) will produce an output of 1, the 2nd uncontrolled (+2) will produce an output of 0, and the 3rd uncontrolled (+2) will produce an output of 2, which will open the 3rd controlled (+2) and will provide the output of 2. Thus, $|D_2>$ will open. Others will remain closed as they will not produce the output of 2. Table 5 illustrates all of the execution. Nine multivalued quantum AND (D_0 to D_8) procedures and two multivalued 1:3 decoders were implemented in the design of the multivalued quantum 2:9 decoder, a combinational logic circuit demonstrated in Figure 11.

Table 5: Quantum Ternary 1:3 Decoder

A	(1) +2	(3) +2	(5) +2	(2) +2 $[D_0]$	(4) +2 $[D_1]$	(6) +2 $[D_2]$
0	2	1	0	2	0	0
1	0	2	1	0	2	0
2	1	0	2	0	0	0

Two inputs, |A> and |B>, and nine outputs, |D_0> to |D_8>, are used for the multivalued Quantum 2:9 decoder.

4. PROPOSED DESIGN OF MPROM IN QUANTUM COMPUTING

To construct the ternary quantum 9:2 PROM, a 2-to-9 decoder is required, along with the minterms of the decoder output serving as inputs to the OR gates to generate the intended QMPROM outputs F1 and F2 according to the Table 6. The following phase describe the design (Figure 12) process of the proposed system:

1. Take |A> and |B>, the two input qubits. The states |0>, |1>, and |2> are the three potential states for each. Nine combinations of the two input qubits will result from each state.
2. 1:3 ternary decoder is required for each input to select alternate combinations of input values. For instance, three values can be executed as |A_0>, |A_1>, and |A_2> for input |A>.
3. For inputs A and B, afterward the two 1:3 ternary decoder operations, the results will be |A_0>, |A_1>, |A_2>, and |B_0>, |B_1>, |B_2>, respectively.
4. It's required to execute AND operations on each of the step 3 values provided to generate 9 several combinations of the 2 input patterns and trigger any output line.
5. Execute ternary OR with (|D_0>, |D_1>), (|D_2>, |D_3>), (|D_4>, |D_5>), (|D_6>, |D_7>) from every combination of ternary AND in Step 4.
6. To produce the output F1 of the ternary 9:2 QPROM, integrate the qubits from Step 5 alongside the ternary OR operation (|D_0>, |D_1>, |D_2>, |D_3>).
7. Afterward, integrate (|D_4>, |D_5>, |D_6>, |D_7>, |D_8>) to produce the other function F2.

- ### Working Principle of Multiple-Valued QPROM

[1] Qubits combination |A>, |B>= |0>, |0>, where |A>= |A_0>, |A_1>, |A_2> = |2>, |0>, |0> and |B> = |B_0>, |B_1>, |B_2>= |2>, |0>, |0>. Consequently, the |A_0> and |B_0> are linked to |D_0>; therefore, |D_0> produces |2>, while all the rest of the gates generate |0>. At this point, the following steps are required to operate output functions.

i. |D_0>, |D_1> = |2>, |0>, the output qubit is |2> since the ternary OR will produce its maximum on the input qubits.
ii. |D_2>, |D_3> = |0>, |0>, the output qubit is |0> since the ternary OR will produce its maximum on the input qubits.
iii. |D_4>, |D_5> = |0>, |0>, the output qubit is |0> since the ternary OR will produce its maximum on the input qubits.
iv. |D_6>, |D_7> = |0>, |0>, the output qubit is |0> since the ternary OR will produce its maximum on the input qubits.
v. After combining qubits |2> and |0> from [i] and [ii], use these as inputs to the ternary OR operations to produce output |2>.
vi. Again, combining qubit |0> from [iii] with [iv], as input to the ternary OR operations to produce output |0>.
vii. Finally, combining qubits |0> from [vii] and D8 (|0>) to ternary OR operations to produce output |2> (v) for |F1> and |0> (vii) for |F2>.

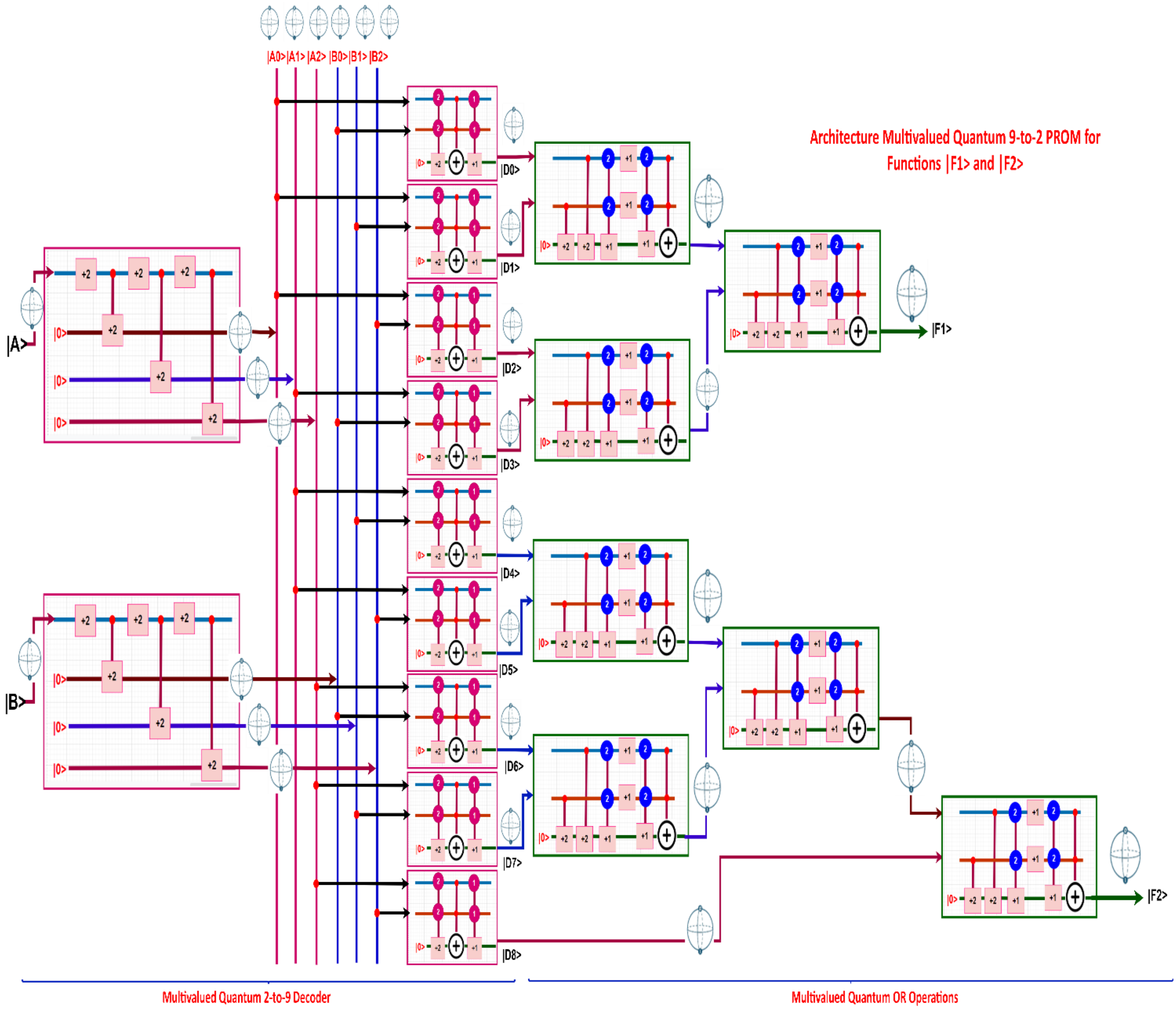

Figure 12: Multivalued Quantum-based 9:2 PROM

[2] Qubits combination |A>, |B>= |0>, |1>, where |A>= |A_0>, |A_1>, |A_2> = |2>, |0>, |0> and |B> = |B_0>, |B_1>, |B_2>= |0>, |2>, |0>. Consequently, the |A_0> and |B_1> are linked to |D_1>; therefore, |D_1> produces |2>, while all the rest of the gates generate |0>. At this point, the following steps are required to operate output functions. Perform OR operations on (|D_0> - |D_3>) outputs to obtain the |F1> value. Therefore, max (|D_0>, |D_1>, |D_2>, |D_3>) = max (|0>, |2>, |0>, |0) = |2>, and for |F2> execute OR operations among (|D_4>, |D_5>, |D_6>, |D_7>, |D_8>) = max (|0>, |0>, |0>, |0>, |0>) = |0>.

[3] Qubits combination |A>, |B>= |0>, |2>, where |A>= |A_0>, |A_1>, |A_2> = |2>, |0>, |0> and |B> = |B_0>, |B_1>, |B_2>= |0>, |0>, |2>. Consequently, the |A_0> and |B_1> are linked to |D_2>; therefore, |D_2> produces |2>, while all the rest of the gates generate |0>. At this point, the following steps are required to operate output functions. Perform OR operations on (|D_0> - |D_3>) outputs to obtain the |F1> value. Therefore, max (|D_0>, |D_1>, |D_2>, |D_3>) = max (|0>, |0>, |2>, |0) = |2>, and for |F2> execute OR operations among (|D_4>, |D_5>, |D_6>, |D_7>, |D_8>) = max (|0>, |0>, |0>, |0>, |0>) = |0>.

[4] Qubits combination |A>, |B>= |1>, |0>, where |A>= |A_0>, |A_1>, |A_2> = |0>, |2>, |0> and |B> = |B_0>, |B_1>, |B_2>= |2>, |0>, |0>. Consequently, the |A_1> and |B_0> are linked to |D_3>; therefore, |D_3> produces |2>, while all the rest of the gates generate |0>. At this point, the following steps are required to operate output functions. Perform OR operations on (|D_0> - |D_3>) outputs to obtain the |F1> value. Therefore, max (|D_0>, |D_1>, |D_2>, |D_3>) = max (|0>, |0>, |2>, |0) = |2>, and for |F2> execute OR operations among (|D_4>, |D_5>, |D_6>, |D_7>, |D_8>) = max (|0>, |0>, |0>, |0>, |0>) = |0>.

[5] Qubits combination |A>, |B>= |1>, |1>, where |A>= |A_0>, |A_1>, |A_2> = |0>, |2>, |0> and |B> = |B_0>, |B_1>, |B_2>= |0>, |2>, |0>. Consequently, the |A_1> and |B_1> are linked to |D_4>; therefore, |D_4> produces |2>, while all the rest of the gates generate |0>. At this point, the following steps are required to operate output functions. Perform OR operations on (|D_0> - |D_3>) outputs to obtain the |F1> value. Therefore, max (|D_0>, |D_1>, |D_2>, |D_3>) = max (|0>, |0>, |0>, |0)

= |0>, and for |F2> execute OR operations among (|D_4>, |D_5>, |D_6>, |D_7>, |D_8>) = max (|2>, |0>, |0>, |0>, |0>) = |2>.

Table 6: Truth Table of Ternary 9:2 QPROM

|A>	|B>	|F1>	|F2>
|0>	|0>	|2>	|0>
|0>	|1>	|2>	|0>
|0>	|2>	|2>	|0>
|1>	|0>	|2>	|0>
|1>	|1>	|0>	|2>
|1>	|2>	|0>	|2>
|2>	|0>	|0>	|2>
|2>	|1>	|0>	|2>
|2>	|2>	|0>	|2>

[6] Qubits combination |A>, |B>= |1>, |2>, where |A>= |A_0>, |A_1>, |A_2> = |0>, |2>, |0> and |B> = |B_0>, |B_1>, |B_2>= |0>, |0>, |2>. Consequently, the |A_1> and |B_2> are linked to |D_5>; therefore, |D_5> produces |2>, while all the rest of the gates generate |0>. At this point, the following steps are required to operate output functions. Perform OR operations on (|D_0> - |D_3>) outputs to obtain the |F1> value. Therefore, max (|D_0>, |D_1>, |D_2>, |D_3>) = max (|0>, |0>, |0>, |0) = |0>, and for |F2> execute OR operations among (|D_4>, |D_5>, |D_6>, |D_7>, |D_8>) = max (|0>, |2>, |0>, |0>, |0>) = |2>.

[7] Qubits combination |A>, |B>= |2>, |0>, where |A>= |A_0>, |A_1>, |A_2> = |0>, |0>, |2> and |B> = |B_0>, |B_1>, |B_2>= |2>, |0>, |0>. Consequently, the |A_2> and |B_0> are linked to |D_6>; therefore, |D_6> produces |2>, while all the rest of the gates generate |0>. At this point, the following steps are required to operate output functions. Perform OR operations on (|D_0> - |D_3>) outputs to obtain the |F1> value. Therefore, max (|D_0>, |D_1>, |D_2>, |D_3>) = max (|0>, |0>, |0>, |0) = |0>, and for |F2> execute OR operations among (|D_4>, |D_5>, |D_6>, |D_7>, |D_8>) = max (|0>, |0>, |0>, |2>, |0>) = |2>.

[8] Qubits combination |A>, |B>= |2>, |1>, where |A>= |A_0>, |A_1>, |A_2> = |0>, |0>, |2> and |B> = |B_0>, |B_1>, |B_2>= |0>, |2>, |0>. Consequently, the |A_2> and |B_1> are linked to |D_7>; therefore, |D_7> produces |2>, while all the rest of the gates generate |0>. At this point, the following steps are required to operate output functions. Perform OR operations on (|D_0> - |D_3>) outputs to obtain the |F1> value. Therefore, max (|D_0>, |D_1>, |D_2>, |D_3>) = max (|0>, |0>, |0>, |0) = |0>, and for |F2> execute OR operations among (|D_4>, |D_5>, |D_6>, |D_7>, |D_8>) = max (|0>, |0>, |0>, |2>, |0>) = |2>.

[9] Qubits combination |A>, |B> = |2>, |2>, where |A> = |A_0>, |A_1>, |A_2> = |0>, |0>, |2> and |B> = |B_0>, |B_1>, |B_2> = |0>, |0>, |2>. Consequently, the |A_2> and |B_2> are linked to |D_8>; therefore, |D_8> produces |2>, while all the rest of the gates generate |0>. At this point, the following steps are required to operate output functions. Perform OR operations on (|D_0> - |D_3>) outputs to obtain the |F1> value. Therefore, max (|D_0>, |D_1>, |D_2>, |D_3>) = max (|0>, |0>, |0>, |0) = |0>, and for |F2> execute OR operations among (|D_4>, |D_5>, |D_6>, |D_7>, |D_8>) = max (|0>, |0>, |0>, |0>, |2>) = |2>. The quantum-based multiple-valued PROM procedure is illustrated by Algorithm 3.

***Algorithm 3**: Quantum-based Multivalued Programmable Read Only Memory (QMPROM)*

Input: |A>, |B>, Output: |F1>, |F2>;
The value of inputs and outputs can be |0> or |1> or |2>

```
Begin
while i = 1 to n do
|P> = DO_ Quant _2to9Decoder ( |Ai>, |Bi> ); // Decoder generates |D0> - |D8>
|P0> <- DO _ Quant_TOR( |D0>, |D1> );
|P1> <- DO _ Quant_TOR( |D2>, |D3> );
|P2> <- DO_Quant_TOR( |D4>, |D5> );
|P3> <- DO_Quant_TOR( |D6>, |D7> );
|P4> <- DO_Quant_TOR( |P2>, |P3> );
|F1> <- DO_Quant_TOR( |P0>, |P1> );
|F2> <- DO_Quant_TOR( |D8>, |P4> );
end while
Procedure DO_Quant_2To9-Decoder( | A >, | B > )
| M > <- DO_Quant_1To3-Decoder( | A > );
| N > <- DO_Quant_1To3-Decoder( | B > );
| D0 > <- DO_Quant_T-AND( | M[0] >, | N[0] > );
| D1 > <- DO_Quant_T-AND( | M[0] >, | N[1] > );
| D2 > <- DO_Quant_T-AND( | M[0] >, | N[2] > );
| D3 > <- DO_Quant_T-AND( | M[1] >, | N[0] > );
| D4 > <- DO_Quant_T-AND( | M[1] >, | N[1] > );
| D5 > <- DO_Quant_T-AND( | M[1] >, | N[2] > );
| D6 > <- DO_Quant_T-AND( | M[2] >, | N[0] > );
| D7 > <- DO_Quant_T-AND( | M[2] >, | N[1] > );
| D8 > <- DO_Quant_T-AND( | M[2] >, | N[2] > );
end procedure
Procedure DO_Quant_T-1To3-Decoder( | A > )
| A2 > <- perform_ PlusTwoOp( | A > );
| D0 > <- perform_ PlusTwoOp( | A2 >, | 0 > );
| A22 > <- perform_ PlusTwoOp( | A2 > );
| D1 > <- perform_ PlusTwoOp( | A22 >, | 0 > );
| A222 > <- perform_ PlusTwoOp( | A22 > );
| D2 > <- perform_ PlusTwoOp( | A222 >, | 0 > );
end procedure Procedure DO_Quant_TOR( | A >, | B > )
| M > <- perform_PlusTwoOp( | B >, | 0 > );
| N > <- perform_PlusTwoOp( | A >, | M > );
| O > <- perform_PlusOneOp2( | A >, | B >, | N> );
| A1 > <- perform_PlusOneOp( | A > );
| B1 > <- perform_PlusOneOp( | B > );
| P > <- perform_PlusOneOp2( |A1>, | B1 >, |O>);
| Q > <- perform_C2NOTOp( | A1 >, | B1 >, |P> );
 end procedure
End
```

5. ANALYSIS OF QMPROM

5.1 Heat Analysis

When qubits in quantum operation become separated and begin to calculate, they produce heat. There is a thermodynamics law in quantum physics, and it applies to qubits in precisely the same form. When there is just one qubit in a quantum system, then a Hamiltonian [36] matrix is formed as: $H = -\frac{1}{2}€\sigma$

In a vertical magnetic field, where € is the energy difference between the states |↑> = |0> and |↓> = |1>, this might be analogous to an electric spin. An atom with two levels, where designate its ground and excited states as |0> and |1>, respectively, may also be referred to by the same Hamilton matrix. The qubit's Gibbs state has the following structure:

$$\rho_{\beta} = \frac{1}{2\cosh\left(\frac{\beta€}{2}\right)} e^{\frac{\beta€\sigma}{2}} = \frac{1}{1+exp\exp(-\beta€)} \left(|0><0| + \right.$$

$e^{-\beta€} |1><1|)$

Consequently, a thermal qubit's average energy,

$$E = \frac{1}{1+ expexp\,(\beta€)}\ , 0 < E < 1/2€$$

This is referred as the thermal qubit's thermodynamic energy. With an eye toward thermodynamics, its possible to compute the von Neumann entropy of the Gibbs state using [36], to describe the r.h.s. using the concept of the energy E.

$$S(E) = -\frac{€-E}{€} log \frac{€-E}{€} - \frac{E}{€} log \frac{E}{€}, 0 < E < 1/2€$$

Then S_{th} Energy will,

$$S_{th}\,(E) = (\,k_B\,ln2\,)\,S(E)$$

Consequently, a single thermal qubit's entropy is, $\frac{dS_{th}(E)}{dE} = \frac{1}{T}$

Furthermore, the n thermal qubit's entropy is, $\frac{dS_{th}(E)}{dE} = \frac{n}{T}$

Knowing that, $\beta = \frac{1}{K_B\,T}$

In this case, T is the starting room and **β** is the inverse temperature, kb is the Boltzmann constant.

$$\beta = \frac{1}{K_B\,T} = \frac{1}{8.617\ X\ 10^{\wedge-5}\ X\ 300} = 39\ \text{evk}^{-1}$$

Thus, the thermal qubit's average energy, $E = \frac{1}{1 \pm e^{39\,X\,\epsilon}} = \frac{1}{1 \pm e^{39\,x\,0.9}} = 5.134 \times 10^{-16}$

In this case, the electron e will have a value of 1.6 ac, and ϵ is emissivity with a value 0 to 1 concerning the molecule. Assume ϵ = 0.9 for ideal purposes.

Now found S_{th} energy as the qubit entropy in quantum mechanics, S_{th} (E) = (Kb ln 2) S(E) = $-k_B \frac{\epsilon - E}{\epsilon} ln \frac{\epsilon - E}{\epsilon} - k_B \frac{E}{\epsilon} ln \frac{\epsilon E}{\epsilon}$ = 178.275 x 10^{-20}

The Multivalued Quantum 9-to-2 PROM is an 8-qubit quantum operation. For the N qubit gate,

S_{th} (E, N) = N (K_B ln 2) S(E/ N). If it is 8 qubits, then N = 8.

$S_{th}\,(E, N) = 2.36179\ x\ 10^{-19}$

$T = \frac{dE\ x\ N}{d\ S_{th}\ (E)} = \frac{5.134\ x\ 10^{-16}\ x\ (8)}{2.36179\ x\ 10^{-19}}$ = 18000.59 k

5.2 Speed Calculation

The theory and formula to determine the average necessary operational time in any quantum computation have been proposed by researchers [37, 38]. Equation 10 computes the average calculation time required for an operation:

$$\tau = \frac{h}{4E}$$

Where τ is the necessary operating time, E is the performing system's quantum mechanical average energy, and h is the plank's constant. Additionally, it has been demonstrated that the following represents the minimal operation time that applies to every digital logic gate used in quantum computation: $\tau = \frac{h}{4E}(1 + 2\frac{\theta}{\pi})$

The phase shift modulo π in this case is θ. It considers any simple quantum gate that adds an arbitrary phase shift to a qubit's state after augmenting it. As many researchers have observed [39], a fundamental quantum operation requires a fundamental amount of time. For example, a single qubit gate operation is required 1μs, a double qubit gate operation is required 10μs, and foe movement operations time is required 20μs. According to these operations, a double qubit gate (CNOT, V, and V+) requires roughly 10μs to operate, while a single gate operation (NOT) requires 1μs. 10μs are needed for a one-bit regulated operation. The C^2NOT gate operation requires 10 + 10 = 20μs of time. The AND gate operation will take 20 + 20 + 20 = 60μs. The multivalued XOR operation takes (20 + 20 + 20 + 1) = 61μs to complete. The multivalued OR operation requires 81μs to complete (10 + 10 + 20 + 1 + 20 + 20). The multivalued NOR operation requires 82μs to complete (10 + 10 + 20 + 1 + 20 + 20 + 1) in total. Since some of the fundamental quantum gate operations are carried out in parallel, required to split the quantum 9-to-2 PROM into pipelines to determine the necessary performing time. The pipelines are: 1) Decoder; AND; OR; OR, and 2) Decoder; AND; OR; OR; OR.

Considering the third pipeline, which is the largest pipeline for processing input to the multivalued quantum 9-to-2 PROM's output, to determine the overall amount of performing time needed. It is evident that the ternary quantum gate operations of AND and OR require 10μs, respectively. The Multivalued Quantum 9:2 PROM requires the following time: = (Decoder + AND + OR + OR + OR) μs, where the fundamental ternary quantum AND operation requires 60μs, the basic ternary quantum OR operation requires 81 μs, and the needed time for the ternary quantum decoder is 13μs. In conclusion, the ternary 3:1 multiplexer requires the following time: (13 + 60 + 81 + 81 + 81) μs = 316μs.

6. CONCLUSION

Quantum computing is still in its early stages, but it promises to transform many areas of science and technology. Ternary quantum circuits can be more efficient for certain types of problems, reducing the number of operations required compared to binary systems. This paper presents the development of quantum-based PROM logic circuits utilizing quantum logic gates, offering a novel pathway in nanoscale computing. The QPROM logic circuits introduced here are designed to enhance circuit compression by leveraging input-dependent compression techniques and reducing a certain quantity of gates through the additional output state being encoded. This quantum ternary-based approach not only advances the efficiency and scalability of storage systems but also opens new possibilities within the field of quantum storage. The ability to process multiple states simultaneously means that quantum computers can explore many possible solutions simultaneously, rather than sequentially. This suggested quantum ternary storage device is poised for potential simulation by quantum computers in an approach not possible with classical computers. Furthermore, compared to conventional or even supercomputers, a quantum computer can work several orders of magnitude faster, making the realization of quantum storage systems through QPROM a promising and practical innovation in quantum computing. As quantum technology evolves, improving time management and heat control will be crucial for realizing practical, large-scale quantum computers.

6. REFERENCES

[1] Schleich, Wolfgang P., et al. "Quantum technology: from research to application." *Applied Physics B,* 122 2016, 1-31.

[2] Marella, Surya Teja, and Hemanth Sai Kumar Parisa. "Introduction to quantum computing." *Quantum Computing and Communications*, 2020.

[3] Horowitz, Mark, and Emily Grumbling, eds. "Quantum computing: progress and prospects." 2019.

[4] Joos, Erich, et al. "*Decoherence and the appearance of a classical world in quantum theory*." *Springer Science & Business Media*, 2013.

[5] Knight, P. L., and B. M. Garraway. "Quantum superpositions in dissipative environments: Decoherence and deconstruction." *Quantum Dynamics of Simple Systems*. CRC Press, 2020. 199-238.

[6] Preskill, John. "Simulating quantum field theory with a quantum computer." *arXiv preprint arXiv:1811.10085*, 2018.

[7] Weigold, Manuela, et al. "Encoding patterns for quantum algorithms." *IET Quantum Communication* 2.4 2021: 141-152.

[8] Dhande, A. P., V. T. Ingole, and V. R. Ghiye. "Ternary digital system: Concepts and applications." 2014.

[9] Khan, Mozammel HA, and Marek A. Perkowski. "Quantum ternary parallel adder/subtractor with partially-look-ahead carry." *Journal of Systems Architecture* 53.7. 2007: 453-464.

[10] Asadi, Mohammad-Ali, Mohammad Mosleh, and Majid Haghparast. "Towards designing quantum reversible ternary multipliers." *Quantum Information Processing* 20.7. 2021: 226.

[11] Glusker, Mark, David M. Hogan, and Pamela Vass. "The ternary calculating machine of Thomas Fowler." *IEEE Annals of the History of Computing* 27.3. 2005: 4-22.

[12] Brusentsov, Nikolay Petrovich, and José Ramil Alvarez. "Ternary computers: The setun and the setun 70." *IFIP Conference on Perspectives on Soviet and Russian Computing*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006.

[13] Heung, Alex, and H. T. Mouftah. "Depletion/enhancement CMOS for a lower power family of three-valued logic circuits." *IEEE Journal of Solid-State Circuits* 20.2. 1985: 609-616.

[14] Pelucchi, Emanuele, et al. "The potential and global outlook of integrated photonics for quantum technologies." *Nature Reviews Physics* 4.3. 2022: 194-208.

[15] Uppu, Ravitej, et al. "Single-photon quantum hardware: towards scalable photonic quantum technology with a quantum advantage." *arXiv preprint arXiv:2103.01110*. 2021.

[16] Heshami, K., England, D.G., Humphreys, P.C., Bustard, P.J., Acosta, V.M., Nunn, J. and Sussman, B.J., 2016. Quantum memories: emerging applications and recent advances. *Journal of modern optics*, 63(20), pp.2005-2028.

[17] Gerhold, M. and Joseph, J., Army Science Planning Strategy Meeting on Integrated Nanophotonics.

[18] Meena, Jagan Singh, et al. "Overview of emerging nonvolatile memory technologies." *Nanoscale research letters* 9. 2014: 1-33.

[19] Baddeley, Alan. "Working memory." *Memory*. Routledge, 2020. 71-111.

[20] Brady, Timothy F., Talia Konkle, and George A. Alvarez. "A review of visual memory capacity: Beyond individual items and toward structured representations." *Journal of vision* 11.5 (2011): 4-4.

[21] Dragojević, Aleksandar, et al. "{FaRM}: Fast remote memory." *11th USENIX Symposium on Networked Systems Design and Implementation (NSDI 14)*. 2014.

[22] Snehi, Jyoti. "Computer peripherals and interfacing." *Firewall Media*, 2006.

[23] Barkalov, Alexander, and Larysa Titarenko. "Logic synthesis for FSM-based control units." Vol. 53. *Berlin: Springer*, 2009.

[24] Kulinich, O., et al. "Modern elementary base of digital systems design." 2023.

[25] Proebster, Walter E. "Digital memory and storage." *Springer-Verlag*, 2013.

[26] Pricer, W. David, et al. "80.1 Integrated Circuits (RAM, ROM)." 2000.

[27] Currie, Edward H. "Mixed-Signal Embedded Systems Design." *Springer International Publishing*, 2021.

[28] Ielmini, Daniele, and Giacomo Pedretti. "Device and circuit architectures for in-memory computing." *Advanced Intelligent Systems* 2.7 2020: 2000040.

[29] Ilyas, Muhammad, Shawn Cui, and Marek Perkowski. "Ternary logic design in topological quantum computing." *Journal of Physics A: Mathematical and Theoretical* 55.30 2022: 305302.

[30] Haghparast, Majid, Robert Wille, and Asma Taheri Monfared. "Towards quantum reversible ternary coded decimal adder." *Quantum Information Processing* 16 2017: 1-25.

[31] Mandal, Sudhindu Bikash, Amlan Chakrabarti, and Susmita Sur-Kolay. "Synthesis techniques for ternary quantum logic." *2011 41st IEEE International Symposium on Multiple-Valued Logic*. IEEE, 2011.

[32] Lin, Sheng, Yong-Bin Kim, and Fabrizio Lombardi. "CNTFET-based design of ternary logic gates and arithmetic circuits." *IEEE transactions on nanotechnology* 10.2 2009: 217-225.

[33] Heung, Alex, and H. T. Mouftah. "Depletion/enhancement CMOS for a lower power family of three-valued logic circuits." *IEEE Journal of Solid-State Circuits* 20.2 1985: 609-616.

[34] Hidary, Jack D., and Jack D. Hidary. "Quantum computing: an applied approach." Vol. 1. Cham: Springer, 2019.

[35] Sandhie, Zarin Tasnim, Farid Uddin Ahmed, and Masud H. Chowdhury. "Background and Future of Multiple Valued Logic." *Beyond Binary Memory Circuits: Multiple-Valued Logic*. Cham: Springer International Publishing, 2022. 1-13.

[36] Diósi, Lajos."A short course in quantum information theory: an approach from theoretical physics." Vol. 827. Springer, 2011.

[37] Levitin, Lev B., Tommaso Toffoli, and Zachary Walton. "Operation time of quantum gates." *arXiv preprint quant-ph/0210076*. 2002.

[38] Isailovic, Nemanja, et al. "Interconnection networks for scalable quantum computers." *ACM SIGARCH Computer Architecture News* 34.2 2006: 366-377.

[39] Thaker, Darshan D., et al. "Quantum memory hierarchies: Efficient designs to match available parallelism in quantum computing." *ACM SIGARCH Computer Architecture News* 34.2 2006: 378-390.