Quantum computing concepts are described using geometric algebra, without using complex numbers or matrices. This novel approach was developed in the first author's Ph.D. dissertation in Electrical Engineering at University of Texas at Dallas (May 2002) and enables the expression of the principle ideas of quantum computation without requiring an advanced degree in mathematics or physics. Using a topologically derived algebraic notation that relies only on addition and the anticommutative geometric product, this paper describes the following quantum computing concepts: bits, vectors, states, orthogonality, qubits, classical states, superposition states, spinor, reversibility, unitary operator, singular, entanglement, ebits, separability, information erasure, destructive interference and measurement. These central quantum concepts can be described simply in geometric algebra, thereby facilitating the understanding of quantum computing concepts by non-physicists and non-mathematicians.

Keywords: Geometric algebra, qubits, ebits, spinors, quantum operators, sparse invariant states, co-occurrence, co-exclusion

Communicated by:

1. Introduction

Quantum computing has received significant attention since the announcement of Shor’s algorithm [1], which demonstrates that quantum computers can solve some extremely computationally intensive problems more efficiently than any classical algorithm.

---

1 Lawrence Technologies LLC, 5485 Beltline Road, Suite 200, Dallas, Texas 75254
Unfortunately, hardware and software engineering for quantum computers requires different sets of skills from either research on the physics of quantum computing or hardware/software engineering for traditional computers. The goal of this paper is to lay a foundation for hardware/software design for quantum computers that is accessible to traditional engineers and computer scientists.

For newcomers to quantum computing the learning curve is steep for two primary reasons. First, quantum computing is based on the principles of quantum physics and is typically expressed mathematically using complex Hilbert space, which is a high-dimensional, complete, vector space, using complex numbers and matrices. The matrix notation is concise and compact, but also opaque to non-mathematicians. Second, quantum computation has many new information concepts that do not naturally arise in classical computing and are therefore unintuitive to traditionally trained engineers and programmers. The difficulty of understanding these new concepts is compounded by the use of Dirac’s “bra-ket” notation [2], since the reader must first comprehend the foreign-looking mathematical notation. This article takes the approach of focusing on quantum computing concepts while relying on the notationally simpler geometric algebra [3,4], which uses neither explicit complex numbers nor matrices.

This article is targeted at engineers and programmers with a basic understanding of computer science and mathematics who are interested in learning about quantum computing. From this perspective, quantum computing is nothing more than an information system with very particular “bit” properties and the approach of this relatively short article is to show the design of a mathematically oriented process structure that naturally represents and models these properties. The key bit and quantum-properties and their relatively simple mathematical representation using geometrical algebra will be introduced when required and only as needed. Bits form the building blocks of the computing industry and computer professionals have very strong intuitions about them, so this article begins with that perspective.

2. Bits Represented as Vectors

A bit expresses a binary distinction, the smallest unit of information, and is physically the space reserved (or bit capacity in a disk, memory, register, or communications channel) for a single binary state value. The typical choice is to use the implementation-specific values 0/1 to symbolically represent mutually exclusive state pairs such as False/True, dark/light, or male/female. Each binary-valued bit is usually given a symbolic name such as $a$ or $b$ to facilitate describing how multiple bit-states causally interact (i.e. $c = \text{NOT } a$, $d = a \text{ AND } b$, using the Boolean algebra conventions with the standard logical operators NOT, AND, OR, etc). A classical bit can only have two possible complementary states and most importantly, these states are required to be mutually exclusive. For example, if states $a = \text{True}$ and $\text{NOT } a = \text{False}$, then bit $a$ cannot simultaneously be both True and False. Multiple bits can be concatenated to express $N = 2^n$ unique states, where $n$ is typically 8 (a byte), 16 or 32 bits. The $N$ resulting states are also mutually exclusive.

The defining properties of classical bits (i.e. $a$, $b$, $c$) are: 1) as above, the complementary state pair is mutually exclusive and 2) the state of each bit can be independently changed. These precise properties can be mathematically represented using vectors (i.e. $a$, $b$, $c$ - in bold font), where each bit is denoted by a distinct vector. This simple choice of representing bits as vectors has many formal mathematical consequences that will be described in footnotes so as
not to disrupt the flow of the article. Two *orthonormal* vectors (orthogonal\(^1\) and unit length) are graphically displayed in Figure 1 as a horizontal and a vertical line that define a plane.

![Figure 1. Two orthonormal vectors \(a\) and \(b\)](image)

A binary *state* is represented here using the ± *orientation* or *direction* of the bit vector and *not* its length, which is always one. Using vector \(a\), bit state \(a = True\) can therefore be denoted as \(+a\) (orientation +1) and NOT \(a = False\) can be denoted as \(-a = \overline{a}\) (orientation –1). The scalar orientation coefficient \(c\) preceding the vector \(ca\) can have the real values of \(c = +1, -1,\) or 0, which naturally leads to a *ternary* state system (similar to tristate logic) with symmetric binary states of \(+1 = +\) and \(-1 = –\), whereas \(0a = 0\) indicates vector \(a\) has no presence. The choice of mapping bits/states into vectors/orientations defines a binary representation that is a *formal linear system* and can be shown to be Boolean complete \([4]\). A linear representation is important when building such a bridge between computer science and physics \([5]\).

Table 1 shows how to define the traditional “addition” operator, denoted as +, for this linear algebraic system. The addition of two vectors can be visualized as the address of a point in the plane of Figure 1, but the vector orientation coefficients follow the usual scalar addition rules in Table 1. This algebra is limited to the set of unit scalar values \(\{0,+1,–1\}\), because this limited scalar set is sufficient to express all necessary distinctions. Table 1 represents modulo 3 addition because repeatedly adding +1 produces the sequence of values \(0 \Rightarrow +1 \Rightarrow –1 \Rightarrow 0\). This choice is isomorphic to the modulo 3 set of values \(\{0, 1, 2\}\) but is symmetric around 0. Addition of any elements in the algebra always produces another element in the algebra.

<table>
<thead>
<tr>
<th>(a + b)</th>
<th>(b = 0)</th>
<th>(b = +1)</th>
<th>(b = –1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a = 0)</td>
<td>0</td>
<td>+1</td>
<td>–1</td>
</tr>
<tr>
<td>(a = +1)</td>
<td>+1</td>
<td>–1</td>
<td>0</td>
</tr>
<tr>
<td>(a = –1)</td>
<td>–1</td>
<td>0</td>
<td>+1</td>
</tr>
</tbody>
</table>

Using Table 1, an important property\(^{ii}\) for addition is \(a + a = –a = \overline{a}\). This codifies the existence of an additive inverse, *complement*, or *negation* state for each state in the algebra. Mutual exclusion of two complementary states can now be expressed as \(a + \overline{a} = 0\), which means the vector \(a\) can only point in one direction at a time because the value 0 has the special meaning of *cannot occur*. The symmetric +/– states naturally describe the *destructive*

\(^{i}\) With \(* = inner product; a\cdot b = 0\) means \(a\) and \(b\) are orthogonal and \(a\cdot a = 1\) means collinear
\(^{ii}\) Due to modulo 3 arithmetic then \(2a = –a\) because \(2a = a + a = –a\) therefore \(2 = –1\) and likewise \(a/2 = a/1 = –a\)
interference of bit vectors, which is critical for quantum computing. Ternary logic is different from traditional Boolean algebra (states of 0/1) because the latter has no third state, and hence confounds the states “the opposite of one” and “nothing”. Since addition is commutative \((a + b = b + a)\) but subtraction is not \((a - b \neq b - a)\), we use the convention that the sign of the coefficient must always be associated with the particular element, for example, \(a - b = a + \overline{b} - \overline{a} + a = -b + a\).

The interpretation of 0 to mean cannot occur [6] is subtle, yet conceptually meaningful, and has these consequences. First, since 0 means cannot occur then a scalar multiplication of state vector by zero, such as \(0a = 0\), simply means that the vector 

\(a\) does not occur or exist and can be removed without any additive effect on an expression (i.e. \(x + 0a = x\)). Therefore, the highlighted cells in Table 1 focus on the addition rules to input states with only the non-zero binary values. These cells can be summarized as the rule: like states invert and differing states pair-wise cancel. Second, assigning a state equation to 0 and then solving for the roots determines the orientation values of states that cannot occur thereby representing the non-solutions of the system, which is the opposite of the conventional meaning. Third, in order for two vectors to exactly cancel, they must be simultaneous, so addition means concurrency in time. This interpretation is consistent with the non-causal nature of quantum computing states. Addition of states is called a co-occurrence [6] because it is impossible to distinguish between (or count) two identical tokens unless they are presented exactly concurrently. Two such identical tokens presented together represent 1 bit of information because it is impossible to know how many truly exist when presented sequentially [6]; {q.v.} chapter 4.

With this brief groundwork in place for classical bits, mutually exclusive states, state inversion and the addition operator (with its interpretation of concurrency), the following section introduces how to represent a qubit (or quantum bit) plus the other properties required to change the qubit state.

3. A Qubit Represented as the Sum of Two Vectors

A qubit requires four states rather than the two states represented by a classical bit, yet still represents only one classical bit because the vectors are constrained to be redundantly encoded. Therefore, a minimum of two classical bit vectors \(\{a_0, a_1\}\) must be used to represent those four possible states. Since the two distinct and orthonormal bit vectors must both simultaneously be allowed to have any binary value, the obvious proposal for a qubit uses addition with all possible non-zero vector orientations:

\[
\text{Qubit: } A = \pm a_0 \pm a_1
\]

There are four possible variations of signs for this sum and they are assigned the state labels \(A_0 = +a_0 -a_1, A_1 = -a_0 +a_1, A_- = -a_0 -a_1, \) and \(A_+ = +a_0 +a_1\), whose meaning will soon be obvious. Similar to the process used for a single vector, we can show that \(A_0 + A_1 = 0\) or \(A_0 = -A_1\) because \((a_0 - a_1) + (-a_0 + a_1) = 0\), which means that states \(A_0\) and \(A_1\) are mutually exclusive. Likewise, states \(A_+\) and \(A_-\) are mutually exclusive, because \(A_+ + A_- = 0\) or \(A_+ = -A_-\). As with \(A_0\) and \(A_1\), the states \(A_+\) and \(A_-\) are pair-wise collinear with the origin (and later these two sets themselves are shown to be orthogonal). Table 2 follows directly from Table 1 when all possible combinations of non-zero orientation values are analyzed (only highlighted cells in Table 1). It is convenient to think informally of this table as the non-solutions from solving \(A_0 = 0\), where the vector coefficients destructively cancel.
Table 2. Valid qubit states highlighted for $\pm a_0 \pm a_1$

<table>
<thead>
<tr>
<th>Row (k)</th>
<th>(a_0)</th>
<th>(a_1)</th>
<th>(A_1 = \overline{a_0} + a_1)</th>
<th>(A_0 = \overline{a_0} + \overline{a_1})</th>
<th>(A_+ = a_0 + a_1)</th>
<th>(A_- = a_0 + \overline{a_1})</th>
</tr>
</thead>
<tbody>
<tr>
<td>(R_0)</td>
<td>(-)</td>
<td>(-)</td>
<td>0</td>
<td>0</td>
<td>+</td>
<td>–</td>
</tr>
<tr>
<td>(R_1)</td>
<td>(-)</td>
<td>(+)</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>0</td>
</tr>
<tr>
<td>(R_2)</td>
<td>(+)</td>
<td>(-)</td>
<td>0</td>
<td>0</td>
<td>+</td>
<td>–</td>
</tr>
<tr>
<td>(R_3)</td>
<td>(+)</td>
<td>(+)</td>
<td>0</td>
<td>0</td>
<td>–</td>
<td>+</td>
</tr>
</tbody>
</table>

Binary combinations of input states

- Anti-symmetric sums are classical states
- Symmetric sums are superposition states

\[ A_1 = R_1 - R_2 \]
\[ A_0 = R_2 - R_1 \]
\[ A_+ = R_0 - R_3 \]
\[ A_- = R_3 - R_0 \]

The four main rows of Table 2 show all the non-zero binary combinations of the orientations for vectors \(\{a_0, a_1\}\). The four right columns show the possible expressions (of symmetric and anti-symmetric sums) with the non-zero or valid states highlighted. The vector and state names were chosen to represent the particular spin properties of the qubit, which acts like a redundantly coded classical bit with complementary states \(A_0 = \overline{A}_1\). State \(A_0\) is selected when coefficient \(c_0\) for vector \(a_0\) is \(c_0 = +\) and state \(A_1\) when the coefficient \(c_1\) for \(a_1\) is \(c_1 = +\), where \(c_0 = -c_1\). Because of these properties \(A_0\) and \(A_1\) are called the classical states of the qubit. Similarly, state \(A_+\) is defined when vector coefficients \(c_0 = c_1 = +\) and state \(A_-\) when vector coefficients \(c_0 = c_1 = -\), thereby representing the superposition states (where \(A_- = \overline{A}_+\)). Figure 2 graphically illustrates these redundantly coded vector and state relationships.

- Superposition States have same signs
- Classical States have opposite signs

Figure 2. Vectors and States for qubit \(A = \pm a_0 \pm a_1\)

It is evident from Figure 2 that the two pairs of states \(\{A_0, A_1\}\) and \(\{A_-, A_+\}\) are compound states that can be represented as a vector (or line) thru the origin, but at a 45 degree angle to either axis. Therefore the sum of vectors also acts like a redundantly encoded vector, because it represents two complementary states. Because of the redundancy, there is more than one way to represent the one classical bit’s worth of states in a qubit. Since the two pairs of states in Figure 2 are 90 degrees apart (which means orthogonal), they are called out of phase representation choices. From the physics perspective, state \(a_0\) is the spin up state (and \(\overline{a_0}\) means NOT \(a_0\)) while state \(a_1\) is the spin down state (and \(\overline{a_1}\) means NOT \(a_1\)). The two classical states \(\{A_0, A_1\}\) represent a symmetrical spinning top pointing up or down. The two superposition states \(\{A_-, A_+\}\) act like a horizontal gyroscopic top supported on one end, so is simultaneously in both/neither of the up/down states. In quantum computing, each spin state is represented as a vector, whereas in classical computing each bit is represented as a vector.
The next topic is the operator that switches between classical and superposition states or phases, which requires multiplication. Table 3 defines the conventional scalar multiplication table for the preceding ternary values. The terms multiplication and product are overloaded in physics and mathematics, because products exist not only for scalars, but also for vectors: the inner products, outer products, tensor products and cross products. Not to be outdone, the primary multiplication operator of geometric algebra is called the geometric product. The geometric product of a state and an operator (applied on the right) produces a new state, where both states and operators are geometric algebra expressions.

Table 3. Scalar Multiplication table for $a * b = b * a$

<table>
<thead>
<tr>
<th>$a * b$</th>
<th>$b = 0$</th>
<th>$b = +1$</th>
<th>$b = −1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a = 0$</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$a = +1$</td>
<td>0</td>
<td>+1</td>
<td>−1</td>
</tr>
<tr>
<td>$a = −1$</td>
<td>0</td>
<td>−1</td>
<td>+1</td>
</tr>
</tbody>
</table>

Scalar multiplication is straightforward\(^1\) and the highlighted cells in Table 3 represent the non-zero binary combinations of the vector orientations. Those cells are equivalent to the XNOR (exclusive NOR) logic behavior, which is summarized as: like states produce +1 and differing states produce −1. XOR/XNOR based logic is identical to the odd/even parity operators and a direct result of the multiplication operator being related to XOR is the unexpected multiplicative inverse property: $1/a = a$ (when $a ≠ 0$). This property is true for both scalars and vectors. As will be shown next, vector multiplication is slightly more complicated in geometric algebra but this complexity enables much simplicity elsewhere.

### 3.1. Geometric Product and Graded N-vectors

The geometric product can now be defined for the multiplication of vectors. As the name implies, the geometric product is based on topological principles. The first simple premise is that multiplying two vectors $(a \ b)$ produces an area-like object called a bivector, which is a different mathematical object type than either a scalar or a vector. Multiplying three vectors together $(a \ b \ c)$ produces a volume-like object called a trivector. This is easy to understand by realizing that a scalar is a grade-0 object (denoted as $A^0$), a vector is a grade-1 object $A^1$, a bivector is a grade-2 object $A^2$, a trivector is a grade-3 object $A^3$, and in general an n-vector is a grade-n object $A^n$ that defines an n-volume. Adding different grade objects creates a multivector of the form; $A = \langle A \rangle_0 + \langle A \rangle_1 + \langle A \rangle_2 + ... + \langle A \rangle_n$. A geometric algebra $G_n$ spanned by n orthonormal vectors contains $N = 2^n$ unique graded elements found by expanding the expression $(1+a)(1+b)...$ and defines $3^N$ unique multivectors, (i.e. $G_2 \Rightarrow 3^4 = 81$). In our definition, a qubit is a multivector: the sum of both grade-1 vectors in $G_2$.

Any bivector has an orientation coefficient just as a vector expression, but with the unusual geometric product identity $a \ b = −b \ a$. This property means that the geometric product is not commutative (more precisely, it is anti-commutative) and is simply the algebraic expression of

---

\(^1\) Scalar multiplication is naturally closed over the ternary values {0, −1, +1}
the right-hand rule used in physics. The bivector orientation coefficient can be imagined as
the right-hand thumb pointing to the front (or back) of a plane defined by a piece of paper and
is depicted in Figure 3. The orientation is defined as the coefficient of any n-vector product in
any grade space\(^7\), so is equivalent to the parity of the vectors of the n-vector in a particular
order. This article places the vectors in standard alphabetically sorted order. Since the
geometric product\(^8\) does not have an explicit operator, writing the product \((\mathbf{a} \mathbf{b})\) therefore
means \((\mathbf{a} \text{ GP} \mathbf{b})\), where the parentheses are optional.

![Figure 3. Geometric product is anticommutative](image)

3.2. Geometric Product and Spinor Operator

The following examples demonstrate the anticommutative geometric product. Assume that a
system is defined or spanned by a set of orthonormal vectors: \(G_2 = \text{span}\{\mathbf{a}, \mathbf{b}\}\). Now multiply
vector \(\mathbf{a}\) times bivector \((\mathbf{a} \mathbf{b})\) and use the topological simplifications\(^9\): \(\mathbf{a} = \mathbf{b} = +1\).

\[
\mathbf{a} (\mathbf{a} \mathbf{b}) = \mathbf{a} \mathbf{a} \mathbf{b} = \mathbf{b}
\]

Similarly, multiply vector \(\mathbf{b}\) times bivector \((\mathbf{a} \mathbf{b})\) and then repeatedly multiply result by \((\mathbf{a} \mathbf{b})\):

\[
\begin{align*}
\mathbf{b} (\mathbf{a} \mathbf{b}) &= \mathbf{b} \mathbf{a} \mathbf{b} = -\mathbf{a} \mathbf{b} = -\mathbf{a} \\
-\mathbf{a} (\mathbf{a} \mathbf{b}) &= -\mathbf{a} \mathbf{a} \mathbf{b} = -\mathbf{b} \\
-\mathbf{b} (\mathbf{a} \mathbf{b}) &= -\mathbf{b} \mathbf{a} \mathbf{b} = +\mathbf{a} \mathbf{b} = \mathbf{a}
\end{align*}
\]

As graphically depicted in Figure 4, the repeated geometric product application of the
bivector \((\mathbf{a} \mathbf{b})\) spins any state counter-clockwise and explains why the bivector \((\mathbf{a} \mathbf{b})\) is
referred to as a spinor. Multiplying by the bivector \((-\mathbf{a} \mathbf{b})\) spins the states in the clockwise
direction, following the right-hand rule. The various qubit encodings are rotations of each
other, so these states are rotationally invariant. You can now relax, since the spinor idea is the
most difficult piece of physics and related mathematics in this article.

The result of a vector multiplied twice by a bivector inverts that vector, which can be
analyzed from only the operator perspective by squaring the operator and simplifying.

\[
(\mathbf{a} \mathbf{b})^2 = (\mathbf{a} \mathbf{b})(\mathbf{a} \mathbf{b}) = \mathbf{a} \mathbf{b} \mathbf{a} \mathbf{b} = -\mathbf{a} \mathbf{a} \mathbf{b} \mathbf{b} = -1
\]

\(^1\) Due to the outer product, and equivalent to the vector cross product only in three dimensions
\(^2\) Geometric product of vectors is the sum \(\mathbf{a} \mathbf{b} = \mathbf{a} \cdot \mathbf{b} + \mathbf{a} \wedge \mathbf{b}\) of inner \(\mathbf{a} \cdot \mathbf{b}\) and outer \(\mathbf{a} \wedge \mathbf{b}\) products.
\(^3\) The simplification of \((\mathbf{a} \mathbf{a}) = 1\) means a vector is collinear with itself and this represents the inner product \(\mathbf{a} \cdot \mathbf{a} = 1\)
Figure 4. Bivector \((a \ b)\) spins the state space counter-clockwise

Since \((a \ b)^2 = -1\), therefore the spinor \(S = (a \ b) = \sqrt{-1}\). Because the spinor operator squared is the inverter (or \(S^2 = \text{NOT}\) operator) the spinor operator is referred to as the square root of not: \(S = (a \ b) = \sqrt{\text{NOT}}\). This is topologically easy to understand using Figure 4 and the anticommutative geometric product\(^1\).

3.3. Reversibility, Unitary Operators, Phases and Pauli Operators

The bivector spinor operator \(S_A = (a0 \ a1)\) for a qubit \(A = (\pm a0 \pm a1)\) is simply an even grade operator that switches between the odd grade classical and superposition phases\(^2\) and can be applied to any state. Also the inversion operator \((S_A)^2 = -1\) can be applied to any state. Both the inverter \(-1 \in \langle A \rangle_0\) and the spinor \(S_A \in \langle A \rangle_2\) are even grade operators, but they also have another important property, called reversibility. Classically speaking, a one-to-one mapping of states is often reversible but any many-to-one state mapping is irreversible.

Just as the name suggests, reversibility refers to an operator that can be reversed or undone. Conventional classical computing, with traditional Boolean logic gates, is typically not reversible due to many-to-one state mappings (effectively, the arrival path is lost), which means information is erased and energy is consumed due to this erasure [7]. Classical computation is reversible by using only the 3-input and 3-output reversible Toffoli or Fredkin gates, rather than the conventional irreversible 2-input gates of NAND/NOR.

Reversibility [8] is easy to describe mathematically with the understanding that all operators are implemented as products. Let’s assume a multivector system state \(X\) and a multivector operator \(Y\) forming some new multivector system state \(Z = X \cdot Y\). To undo this operator means convert the state \(Z\) back into state \(X\). This is possible by simply dividing by \(Y\) (or multiplying times \(1/Y = Y^{-1}\)) resulting in \(Z/Y = X\). The operator \(Y\) is reversible if and only if the multiplicative inverse \(W = 1/Y = Y^{-1}\) exists. An operator \(Y\) with this property (i.e. \(1/Y\) exists) is called unitary because \(YW = Y\cdot Y^{-1} = +1\) and this formal definition is semantically synonymous with any reversible operator \(Y\).

\(^1\) Due to these spinor properties, qubits are referred to as possessing spin \(\frac{1}{2}\).

\(^2\) A spinor is the same as the Hadamard operator.
The good news about reversibility is that scalars (1/a = a), vectors (1/a = a), n-vectors (1/S_A = –S_A) and many multivectors (1/A_0 = A_1, 1/A_1 = A_0) are reversible because geometric products are invertible \cite{3}. The term invertible means (to physicists) that expressions have a multiplicative inverse, which is implemented in geometric algebra as the additive inverse (or negation).

A useful multivector example $P_A = -1 + S_A$, is invertible ($1/P_A = 1 - S_A$) and has several other properties. First, $P_A$ is of even grade just like its additive operands. Second, $(P_A)^2 = S_A$ so consequently $\sqrt{S_A} = \pm P_A$. It is now possible to summarize the previously seen discrete phase relationships. $+1 = 360^\circ$, NOT = $\sqrt{+1} = 180^\circ$, spinor = $\sqrt{\text{NOT}} = 90^\circ$, so the square root of an operator (if it exists) is related to dividing the spin angle in half and similarly the third root angle is $360^\circ/3 = 120^\circ$, etc. Third, since the operator $-1$ means inversion and $S_A$ means phase spin, then with our interpretation of addition, $P_A$ is simultaneously an inversion and phase shift! Here are the results of the Pauli operator $P_A$ applied to the four qubit states.

\begin{align*}
A_0 P_A &= A_0 (-1) + A_0 S_A = A_1 + A_1 = (-a_0 + a_1) + (a_0 + a_1) = -a_0 + a_0 + a_1 + a_1 = -a_1 \\
A_1 P_A &= A_0 + A_1 = +a_1 \\
A_- P_A &= A_+ + A_0 = -a_0 \\
A_+ P_A &= A_- + A_1 = +a_0
\end{align*}

The Pauli operator $P_A$ reversibly maps the classical states $A_{1/0}$ to the vertical vector $\pm a_1$ and maps the superposition states $A_{\pm}$ to the horizontal vector $\pm a_0$. As expected and as graphically seen in Figure 5, this represents a discrete 45 degree phase encoding away from the classical and superposition axes. The inversion and spinor operators still function as expected for this representation. This vector encoding \cite{ii} emphasizes the classical/superposition meaning of the vectors rather than the spin up/down meaning, yet both interpretations are valid.

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{Fig5.png}
\caption{Phase Encodings of 180°, 90° and 45° for qubit $A = (\pm a_0 \pm a_1)$}
\end{figure}

All three Pauli operators have now been discussed ($\sigma_i = -1$, $\sigma_0 = S_A$, $\sigma_1 = P_A$). This is important to quantum computing mathematics because the Pauli operators represent the reversible even-grade operators that encode how noise can affect a qubit state as either a bit inversion, a phase shift, or both a bit & phase shift simultaneously. Likewise, the odd-grade reversible operators $\pm a_0$, $\pm a_1$ and $(\pm a_0 \pm a_1)$ also produce alternate encodings to the even

\footnote{Neither the inner nor outer products are invertible by themselves}

\footnote{“Encoding” means basis: classical = standard basis, superposition = dual basis, and circular basis = $\pm 1 \pm S_A$}
grade plane formed by the axes ±1 and ±$\mathbf{S}$A. Quantum computation primarily involves reversibly rotating a qubit encoding through a phase angle without erasing the bit of information stored in the qubit, so all single-qubit operators are specific kinds of phase gates.

Of the total of $3^N-1 = 80$ possible qubit multivectors (excludes state 0), 48 are reversible because they are invertible. The remaining 32 multivectors do not have multiplicative inverses and are thus irreversible. The next section describes how to identify these irreversible operators. Of the 80 possible multivector states, 40 multivectors are the additive inverses of the other 40, and all of these 40 unique states are discussed as sets in this article.

### 3.4. Irreversibility, Singular Operators, Erasure and Measurement

Irreversible operators are important in quantum mechanics because they erase the information encoded in a qubit. Losing information is bad, because the wrong answer will emerge when asking for an answer with a measurement question. This situation is problematic for qubits because noise is equivalent to an unwanted operator. If the system is in an “unexpected” state, then the basis-based question asked (see below) will ipso facto be ill-formed, resulting in a random binary answer from the measurement. Additionally, all measurement operators are irreversible and destroy the qubit state by setting the qubit to the questioned state. Extracting the information stored in a qubit destroys the state, so there is only one chance. This is similar to old core memory systems, which required writing the data back after a destructive read.

The simplest non-invertible multivector has the form $X = (\pm 1 \pm \mathbf{x})$, where $\mathbf{x}$ is any 1-vector. Simply stated, because $X^{-1} = (\pm 1 \pm \mathbf{x})^{-1}$ does not exist, then $X$ is called singular. This fact is the basis for all other singular operators of a qubit because using the product $X Y = Z$, if either factor $X$ or $Y$ is singular then so is $Z$. Alternatively, when $X$ is unitary then $X^{-1}$ exists and the multivector $X$ is non-singular.

All of the 32 singular multivectors of a qubit contain one of the factors $(\pm 1 \pm \mathbf{x})$, and they are: $(\pm 1 \pm \mathbf{a}0) = 4$, $(\pm 1 \pm \mathbf{a}1) = 4$, $\mathbf{a}0(\pm 1 \pm \mathbf{a}1) = 4$, $\mathbf{a}1(\pm 1 \pm \mathbf{a}0) = 4$, $(\pm 1 \pm \mathbf{a}0)(\pm 1 \pm \mathbf{a}1) = 8$ and the opposite order $(\pm 1 \pm \mathbf{a}1)(\pm 1 \pm \mathbf{a}0) = 8$ for a total of $4 + 4 + 4 + 4 + 8 + 8 = 32$ unique singular multivectors. As is shown below, all of these singular operators are related to measurement and information erasure. Each operator $X$ in the list above was proved to be singular by exhaustively attempting to solve the equality $X Y = Z$, for each of the possible 80 multivectors $Y$, and no solutions were found. For expressions involving multiple qubits, other singular expressions exist, however, that do not have $(\pm 1 \pm \mathbf{x})$ as a factor.

Knowing exactly how measurement occurs, answers are extracted, and information is erased, in a qubit is important for quantum computing, and singular operators are an important clue to this understanding. Essentially, a measurement entails asking what state orientation a particular vector currently possesses. In geometric algebra, a multivector of the form $X = (–1)(1 \pm \mathbf{x})$ can be used to isolate only the state cases for orientation $\pm \mathbf{x}$ and so is equivalent to testing or decoding vector $\mathbf{x}$ for a particular orientation, denoted as $X_{\pm}$. Each of the four output columns in Table 4 represents one of the singular expressions of the form $(–1)(1 \pm \mathbf{x})$.

---

1 Identical to real (scalars) and imaginary axes ($\mathbf{S}_A = i$) as represented in complex numbers.
2 Geometric algebra rotators $\mathbf{R} = R \mathbf{e}_a \mathbf{R}^* :$ with $R = \alpha - \beta e_1 e_2$, $\mathbf{R}^* = \alpha + \beta e_1 e_2$, $\alpha = \cos(\theta/2), \beta = \sin(\theta/2)$
3 Formal definition of unitary is $|\det(X)| = +1$, which is true for all non-singular multivectors $X$ if $\det(X) > 0$ [9]
4 $X$ is singular if $\det(X) = 0$ because $X^{-1}$ becomes infinite due to $1/\det(X)$ being dependent on $1/\det(X)$
5 For $X Y = Z$, then $\det(X)\det(Y) = \det(Z)$, so if $\det(X) = 0$ or $\det(Y) = 0$ then $\det(Z) = 0$
Table 4. Specifying a particular vector orientation in \( G_2 = \text{span}\{a_0, a_1\} \).

<table>
<thead>
<tr>
<th>Row ( k )</th>
<th>( a_0 )</th>
<th>( a_1 )</th>
<th>((-1)(1 - a_0))</th>
<th>((-1)(1 + a_0))</th>
<th>((-1)(1 - a_1))</th>
<th>((-1)(1 + a_1))</th>
</tr>
</thead>
<tbody>
<tr>
<td>( R_0 )</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>+</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( R_1 )</td>
<td>0</td>
<td>+</td>
<td>+</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( R_2 )</td>
<td>+</td>
<td>0</td>
<td>0</td>
<td>+</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( R_3 )</td>
<td>+</td>
<td>+</td>
<td>0</td>
<td>+</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Summation of \( R_k \)  
\( A_0 = R_0 + R_1 \)  
\( A_0 = R_2 + R_3 \)  
\( A_1 = R_0 + R_2 \)  
\( A_1 = R_1 + R_3 \)

Denoted as Vector \( [+ + 0 0] \)  
\( [0 0 + +] \)  
\( [+ 0 + 0] \)  
\( [0 + 0 +] \)

In every column, two rows contain the + state and two rows contain the 0 state. When this expression is used as an operator it effectively creates a notch filter that only passes the non-zero states. By combining two orientation choices using the geometric product, a particular row can be selected, which specifies the logically combined state \( A_0 \pm \) and \( A_1 \pm \), so each row \( R_k \) represents a cell in a Boolean logic Karnaugh map \(^1\) used by conventional logic designers.

\[ A_0 \pm A_1 \pm = (1 \pm a_0 \pm a_0 \pm a_0 a_1) \]

\[ A_0 A_1 = (1 - a_0 - a_1 + a_0 a_1) \]

\[ A_0 A_1 = (1 + a_0 - a_1 - a_0 a_1) \]

\[ A_0 A_1 = (1 + a_0 + a_1 + a_0 a_1) \]

(6)

Table 5. Specifying two vector orientations in \( G_2 \).

<table>
<thead>
<tr>
<th>Row ( k )</th>
<th>( a_0 )</th>
<th>( a_1 )</th>
<th>((-1)(1 - a_0)(1 - a_1))</th>
<th>((-1)(1 - a_0)(1 + a_1))</th>
<th>((-1)(1 + a_0)(1 - a_1))</th>
<th>((-1)(1 + a_0)(1 + a_1))</th>
</tr>
</thead>
<tbody>
<tr>
<td>( R_0 )</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( R_1 )</td>
<td>0</td>
<td>+</td>
<td>0</td>
<td>+</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( R_2 )</td>
<td>+</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>+</td>
<td>0</td>
</tr>
<tr>
<td>( R_3 )</td>
<td>+</td>
<td>+</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>+</td>
</tr>
</tbody>
</table>

State logic \( R_0 = A_0 A_1 \)  
\( R_1 = A_0 A_1 \)  
\( R_2 = A_0 A_1 \)  
\( R_3 = A_0 A_1 \)

Denoted as Vector \( R_0 = \begin{bmatrix} 0 & 0 & 0 & 0 \end{bmatrix} \)  
\( R_1 = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix} \)  
\( R_2 = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix} \)  
\( R_3 = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix} \)

Table 5 illustrates these singular expressions, which represent the topologically smallest features in a qubit representation. These row-decode operators, \( R_k \) are linearly independent and all other expressions can be derived by summing specific rows, so each algebraic expression has a unique, dual, sparse representation expressed as the sum of \( R_k \). The inverse of \( R_k \) is denoted as \( P_k = -R_k \). The compact vector-like notation \( \begin{bmatrix} R_0 & R_1 & R_2 & R_3 \end{bmatrix} \) expresses these states, where the row values \( R_k \in \{0, -, +\} \) are the values of the expressions for every non-zero combination of vector orientations. This vector notation can be thought of as a matrix diagonal because \( R_0 + R_1 + R_2 + R_3 = [+ + + +] = +1 \), and \( P_0 + P_1 + P_2 + P_3 = [- - - -] = -1 \). The vector notations for several other familiar multivectors are: \( a_0 = [- - + +] \), \( a_1 = [- + - +] \), \( S_A = [+ - - +] \), \( A_0 = [0 + 0] \), \( A_1 = [0 - 0] \), \( A_+ = [0 0 +] \) and \( P_A = [0 + 0] \).

Element by element vector addition is identical to algebraic addition, for example the sum: \( a_0 + a_1 = [- - + +] + [- + - +] = [+ 0 0 -] = A_+ \), because the \( R_k \) are linearly independent.

---

\(^1\) The vector notation is the set of \( R_k \) denoted as a vector \( [R_0 R_1 R_2 R_3] \) and is used extensively in this paper.

\(^2\) \( R_k \) are the computational basis: different from standard basis since multiplication=XNOR vs. AND in Hilbert space.
The overall qubit singular-operator relationships are now shown in Table 6, which illustrates the answer to measuring the four qubit states (in first column) from the perspective of each singular row-decode operator $R_k = A_0 + A_1$. This table is an example of a set of one-to-one mappings that is irreversible because the mapping operators are singular and so cannot be undone. Classical Boolean logic systems do not have the concept of singular operators.

Table 6. Qubit measurement results for $G_2$

<table>
<thead>
<tr>
<th>Start States $A$</th>
<th>Each start state $A$ times each $R_k$</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A_0 = +a_0 - a_1$</td>
<td>$A(1+a_0)(1-a_1)$ $A(1-a_0)(1+a_1)$ $A(1+a_0)(1+a_1)$ $A(1-a_0)(1-a_1)$</td>
<td>Classical States Measurement</td>
</tr>
<tr>
<td>$A_1 = -a_0 + a_1$</td>
<td>$+1 - a_1 = I^+$ $+1 + a_1 = I^-$ $-a_0 (+1 + a_1)$ $+a_0 (-1 + a_1)$ $-a_0 (-1 - a_1)$ $+a_0 (+1 - a_1)$</td>
<td>Superposition States Measurement</td>
</tr>
<tr>
<td>$A_σ = -a_0 + a_1$</td>
<td>$-a_0 (-1 + a_1)$ $+a_0 (+1 - a_1)$ $+a_0 (-1 + a_1)$ $-a_0 (+1 - a_1)$</td>
<td></td>
</tr>
<tr>
<td>$A_σ = +a_0 + a_1$</td>
<td>$-a_0 (+1 - a_1)$ $+a_0 (-1 + a_1)$ $-a_0 (+1 - a_1)$ $+a_0 (-1 + a_1)$</td>
<td></td>
</tr>
</tbody>
</table>

Applying the singular operators $R_k$, Table 6 produces two kinds of singular answers, either a “sparse invariant” or a random value. The measurement returns the answer and the qubit changes to the end state after measurement. The resulting answers of the form $(\pm 1 \pm a_1) = \pm I$ act like a constant since the non-zero output row-states are either all $+$ or all $-$, as follows. This was originally hinted at in Table 4, where expressions $-1 + a_1 = [+ 0 + 0] = I^-$ and $-1 - a_1 = [0 + 0 +] = I^+$ are two out-of-phase examples of sparse invariants. This name was coined because the multivectors $I^+$ act like sparse versions of the constants $\pm 1$, with the properties $I^- = -I^+$ and $(I^+)^2 = I^+$. The sum of two out-of-phase versions of these invariants form the constants $+1 = I^0_0 + I^+_{99'} = [+ + + +]$ and $-1 = I^-_{09'} + I^-_{99'} = [- - - -]$. Any multivector of the form $(\pm 1 \pm X)$ is a sparse invariant, where $X$ is any n-vector. Not all sparse invariants are singular (e.g. $P_\lambda = -1 + S_\lambda = [0 + + 0]$).

From a measurement perspective, the sparse invariants $I^+$ represent a Boolean answer because the result is $I^+$ or $I^-$, and the qubit is projected to the end state matching the question. This process is irreversible because both $R_k$ and $I^+$ are singular. From the sums of $R_k$ or vector notation, it is easy to see how information is erased because the symmetry of the qubit is broken. The symmetry is essentially based on which rows are valid, where the rows $\{R_1, R_2\}$ are non-zero only for the classical states and the rows $\{R_0, R_3\}$ are non-zero only for the superposition states. The sparse invariants include a row state from each pair of rows $I^+_0 = [+ 0 + 0] = R_0 + R_2$ and $I^-_{99'} = [0 + 0 +] = R_1 + R_3$, so the combined asymmetrical state is no longer linearly independent since it is the sum of non-orthogonal elements.

---

1 Symmetry or coherence, whereas asymmetry means decoherence
2 Pair-wise orthogonal $R_\times R_\times = 0$ are the standard basis and $R_\times R_\times = 0$ are the dual basis.
3 Non-orthogonal vectors cannot be used as the matrix basis vectors for quantum systems.
The row-pair symmetry is also broken by singular operators of the form \( \pm a_0 \pm a_1 a_1 \) because \( a_0 + a_0 a_1 = R_1 - R_3, -a_0 - a_0 a_1 = R_3 - R_1, a_0 - a_0 a_1 = R_0 - R_2, \) and \(-a_0 + a_0 a_1 = R_2 - R_0\). Each of these results looks like a random value because half the states are + and other half are –, or statistically random, in contrast to the invariants, which are all the same value.

The row-decode operators \( R_k = A O_k \) are also asymmetrical since they each contain only one non-zero row.

The above discussion utilizes only half of the singular states of a qubit. Exactly the same analysis can be performed using the anticommutative or dual versions of the row-decode operator products \( R_{7-k} = A I_k A O_k \) (dual of \( R_k = A O_k A I_k \)) These expressions represent the other four multivectors of the form \((1 \pm a_0 \pm a_1 \pm a_0 a_1)\), where the sign is inverted for the bivector, resulting in all zero-valued row-states being converted to the – state.

\[
\begin{align*}
A_{1-k} \Delta_{0} &= (1 + a_0 + a_1 - a_0 a_1) = [+ + - -] = R_7 \text{ where } R_0 = [0 0 0 0] \\
A_{1-k} \Delta_{0} &= (1 - a_0 - a_1 + a_0 a_1) = [- - + +] = R_6 \text{ where } R_1 = [0 0 0 0] \\
A_{1-k} \Delta_0 &= (1 - a_0 + a_1 + a_0 a_1) = [- - + +] = R_5 \text{ where } R_2 = [0 0 0 0] \\
A_{1-k} \Delta_0 &= (1 - a_0 - a_1 - a_0 a_1) = [+ + - -] = R_4 \text{ where } R_3 = [0 0 0 0] \\
\end{align*}
\]

With the inverted operators \( P_{7-k} = -R_{7-k} \) also defined, then the following facts are true about \( R_{1-k} \): \( R_3 + R_5 + R_6 + R_7 = +1 \) and \( P_3 + P_5 + P_6 + P_7 = -1 \). The overall unitarity\(^1\) property of a qubit is defined as \( P_3 + P_5 + P_6 + P_7 = +1 \) and \( R_3 + R_5 + R_6 + R_7 = +1 \). The multivectors in \( C_2 \) with this property\(^2\) have the form of \( E_k = (\pm a_0 \pm a_1 \pm a_0 a_1) \) and represent the eight corners of the cube in Figure 6, formed by the axes \{\pm a_0, \pm a_1, \pm a_0 a_1\}. These multivectors form the corners of the dual tetrahedrons formed by the sides \( P_k = -(1+E_k) \) or \( E_k = R_k -1 \) shown in Figure 7. Even though the axes are drawn in a cube, they are not orthogonal.

An important and interesting topological fact is that these set of eight multivectors have the invertibility property \( X = 1/X = X^{-1} \), and therefore are self-unitary: \( X = X^{-1} = X X = X^2 = 1 \). The multivectors in \( C_2 \) with this property\(^3\) have the form of \( E_k = (\pm a_0 \pm a_1 \pm a_0 a_1) \) and represent the eight corners of the cube in Figure 6, formed by the axes \{\pm a_0, \pm a_1, \pm a_0 a_1\}. These multivectors form the corners of the dual tetrahedrons formed by the sides \( P_k = -(1+E_k) \) or \( E_k = R_k -1 \) shown in Figure 7. Even though the axes are drawn in a cube, they are not orthogonal.

An interesting and important topological fact is that these set of eight multivectors have the invertibility property \( X = 1/X = X^{-1} \), and therefore are self-unitary: \( X = X^{-1} = X X = X^2 = 1 \). The multivectors in \( C_2 \) with this property\(^3\) have the form of \( E_k = (\pm a_0 \pm a_1 \pm a_0 a_1) \) and represent the eight corners of the cube in Figure 6, formed by the axes \{\pm a_0, \pm a_1, \pm a_0 a_1\}. These multivectors form the corners of the dual tetrahedrons formed by the sides \( P_k = -(1+E_k) \) or \( E_k = R_k -1 \) shown in Figure 7. Even though the axes are drawn in a cube, they are not orthogonal.

The results in Figures 6 and 7 are topologically interesting and very relevant to matrix mathematics. One of the important results of the relationships, \( R_k = (1+E_k) \) and \( (E_k)^2 = 1 \) is that the product [10] \( E_k R_k = E_k (1+E_k) = E_k + E_k = 2E_k = 2R_k \), which ultimately leads to the important result\(^4\) that \( P_k P_k = (P_k)^2 = P_k \), where the \( P_k \) form the sides of the dual tetrahedrons in Figure 7.

Table 7 summarizes these multivector relationships including the sum of all \( E_k=0 \).

---

\(^1\) Same as the unitarity constraint for qubits in Hilbert Space

\(^2\) Property \( E_k E_k = 1 \) means the \( E_k \) are the eigenvectors and \( P_k = -(1+E_k) \) are the projection operators

\(^3\) The \( P_k \) are idempotent \( (P_k)^2 = P_k \), projection operators of the qubit, so are the eigenvalues of the eigenvectors \( E_k \).
The symmetric results in Table 7 show that our algebraic notation naturally describes a qubit and is formally equivalent to the matrix notation traditionally used for the same purpose. Even though establishing the foundational concepts of qubits relies on some fairly abstruse mathematics, once these are in place, one need only the relatively straightforward manipulation of geometric algebra to read, write, manipulate, interpret, and understand qubits. Nevertheless, quantum concepts themselves still constitute a relatively steep learning curve.

The last remaining set of expressions from the 80 qubit states is called the trine states. Trines are mathematically easy to identify because they represent the eight solutions of the equality $(Tr)^3 = 1$. The qubit solutions all have the form $Tr = (+1 ± a0 ± S_A)$ or $Tr = (+1 ± a1 ± S_A)$ and their inverses. The general form is the concurrent sum of the spinor and a singular operator of the form $(+1 ± x)$. As expected and as seen in state evolution in Eq. (8), this 120° operator causes the state space to become asymmetrical. These operators are unitary though, because the multivector $Tr$ is invertible since $1/Tr = (Tr)^2$.

$$A_0 = [0 + - 0]$$
$$A_0 (+1 + a\theta + S_A) = (+1 - a\theta + S_A) = [0 + - +]$$
$$A_0 (+1 + a\theta + S_A)^2 = (-1 + a\theta - S_A) = [0 + - -]$$
$$A_0 (+1 + a\theta + S_A)^3 = A_0 = [0 + - 0]$$ (8)

The next section describes combining multiple qubits to form a quantum register.

---

1 For the full table of 40/80 operators see table 7.2 in reference [4].
4. Quantum Registers as Geometric Product of Qubits

Multiple $q$ qubits can be combined to form a quantum register $Q_q = C_{n=2q}$ that defines a space of size $n = 2q$. The state space of two qubits with $n = 4$ does not have the size of $4 + 4 = 8$ states, but rather $N = 2^4 = 16 = 4 \times 4$ total states and $3^{16} = 43,046,721$ discrete multivectors. The number of states grows exponentially because combining qubits entails entangling their state spaces. Geometric algebra easily expresses qubit entanglement using the geometric product. The entanglement of $q = 2$ qubits, defined as $A = (\pm a_0 \pm a_1)$ and $B = (\pm b_0 \pm b_1)$, is simply the geometric product $A B$ of the qubits:

$$A B = (\pm a_0 \pm a_1)(\pm b_0 \pm b_1) = \pm a_0 b_0 \pm a_0 b_1 \pm a_1 b_0 \pm a_1 b_1$$  

(9)

This sum of four bivectors represents all the possible simultaneous combinations of the spin vectors. Recalling the spinor notation for each qubit (i.e. $S_A$, $S_B$, etc), these bivectors are actually cross-qubit spinors and are denoted as $S_{00} = a_0 b_0$, $S_{01} = a_0 b_1$, $S_{10} = a_1 b_0$ and $S_{11} = a_1 b_1$, with all vectors in the standard sorted order. The product of sums format on the left is mathematically identical to the sum of products format on the right. If a sum of bivectors can be factored back into a product of sums format, the entangled states are called separable.

Specific examples with each qubit in specific states produce a vector notation with 16 rows. The number of states grows as $N = 2^{2q} = 4^q$, but the number of non-zero states only grows as $2^q = 4$. Notice that sum of products for $A_0 B_0$ is indistinguishable from $A_j B_j$ so $A_0 B_0 = A_1 B_1$.

$$A_0 B_0 = (a_0-a_1)(b_0-b_1) = +a_0 b_0 - a_0 b_1 - a_1 b_0 + a_1 b_1$$

$$A_0 B_1 = (a_0-a_1)(b_1-b_0) = -a_0 b_0 + a_0 b_1 + a_1 b_0 - a_1 b_1$$

$$A_1 B_0 = (a_1-a_0)(b_0-b_1) = -a_0 b_0 + a_0 b_1 + a_1 b_0 - a_1 b_1$$

$$A_1 B_1 = (a_1-a_0)(b_1+b_0) = +a_0 b_0 + a_0 b_1 + a_1 b_0 + a_1 b_1$$  

(10)

Using the multiplication principle $0 x = 0$, then the valid or non-zero states of both qubits must be satisfied simultaneously. As shown in Table 8, if the 16 row vectors are determined for the above examples, then the valid rows are: $A_0 B_0 = -R_3 + R_6 + R_9 - R_{10}$ and $A_1 B_0 = R_0 - R_3 - R_{12} + R_{15}$ based on the simultaneity constraint that both qubits are contributing non-zero states.

As expected, the valid states of the system are just the valid states for each qubit spread out across a larger space. The green highlighted rows $\{R_6, R_9, R_{10}\}$ indicate the classical states $A_0$ and $B_0$. The blue highlighted rows $\{R_0, R_3, R_{12}, R_{15}\}$ indicate the superposed states $A_1$ and $B_1$. A very interesting intermediate result noted in the rose colored middle columns is an equal number of both orientations. Consequently, all non-zero outputs can occur only when all the bivector orientation coefficients have exactly the same sign. This pair-wise cancellation result is therefore independent of the mod 3 addition conventions established initially. For more examples, discussion and proof see [4].

---

1 $C_{n=3}$ is called a qutrit where multivector state $A = (\pm a_0 \pm a_1 \pm a_2)$ and describes a spin-one particle like a photon.

2 Geometric product is same as tensor product $\otimes$ in Hilbert spaces and tensor power $X^{\otimes n}$ is the power $X^n$.

3 For $Q_k$, the $P_k = -R_k$ are singular, but are idempotent only if the definition is extended to: $(P_k)_{mod3} = P_k$.

4 The number of spinors $s=2^n$ contains only even factors, so $s/3 = \pm 1 \neq 0$, so zero occurs only when $+1-1 = 0$. 

Separable qubits each can be individually manipulated using the appropriate operators, and the operators can be thought of as being sequentially applied, producing various intermediate states. Due to non-commutative products, remember that $A_0 B_0 = -B_0 A_0$ (except for even grade operators that are commutative, such as $S_A = S_A B$).

$$A_0 B_0 S_A = A_0 S_A B_0 = A_0 B_0 = + a_0 b_0 - a_0 b_1 + a_1 b_0 - a_1 b_1$$
$$A_0 B_0 S_B = A_0 S_A B_0 S_B = A_0 B_e = + a_0 b_0 + a_0 b_1 + a_1 b_0 + a_1 b_1$$

Also understand that the Pauli operators applied to both qubits define the cross-qubit spinors.

$$A_0 B_0 P_A P_B = A_0 P_A B_0 P_B = a_1 b_1 = S_{11}$$ and likewise
$$A_1 B.e P_A P_B = a_0 b_0 = S_{00}$$
$$A_e B_1 P_A P_B = a_0 b_1 = S_{01}$$
$$A_1 B_e P_A P_B = a_1 b_0 = S_{10}$$

This implies that the sum of spinor products is identical to representing the qubits in four distinct states simultaneously (i.e. superposed) in the Pauli encoding. In fact, this is exactly the previous meaning of a sum of cross-qubit spinors, since addition means concurrent.

### 4.1. Ebits and Bell States

A very interesting result regarding two qubits is applying both spinors concurrently ($S_A + S_B$) rather than sequentially ($S_A S_B$) to produce an ebit. Half of the bivectors disappear due to destructive interference. As a consequence, this result is inseparable and the reason is the erasure of phase-states. Just as a single qubit is a computational resource due to superposition...
of states, an ebit is also a computational resource because it encodes exactly one classical bit of information (one bit being erased), even if the qubits are separated by a large distance [11]. The ebit’s property is that of an Einstein-Podolsky-Rosen (EPR) communications resource.

\[ A_0 B_0 (S_A + S_B) = A_0 B_0 + A_0 B_x = -a_0 b_0 + 0 a_0 b_1 + 0 a_1 b_0 + a_1 b_1 = -a_0 b_0 + a_1 b_1 \] (13)

This state is one of the four Bell states \(^1\). The concurrent spinor \(B = (S_A + S_B)\), which turns out to be the Bell operator, iteratively generates all four Bell states \((B_0 => B_1 => B_2 => B_3)\) using the formula \(B_{i+1} = B_i B\). Table 8 shows the very interesting result that the only valid states are where exactly one qubit occupies the superposition state at a time. The unlisted rows are zero, so do not occur. This property is also holds true for valid row states for any number of qubits as: \(A_0 B_0 C_0 \ldots (S_A + S_B + S_C + \ldots)\). This symmetry is quite fascinating!

Table 8. Valid rows for ebit \(B_0\) in \(Q_2\)

<table>
<thead>
<tr>
<th>Row</th>
<th>State Combinations</th>
<th>Individual bivectors</th>
<th>Output column</th>
</tr>
</thead>
<tbody>
<tr>
<td>(R_1)</td>
<td>(-) (-) (-) (+)</td>
<td>(-a_0 b_0) (+a_1 b_1)</td>
<td>(+)</td>
</tr>
<tr>
<td>(R_2)</td>
<td>(-) (-) (+) (-)</td>
<td>(+a_0 b_0) (+a_1 b_1)</td>
<td>(-)</td>
</tr>
<tr>
<td>(R_4)</td>
<td>(-) (+) (-) (-)</td>
<td>(-a_0 b_0) (+a_1 b_1)</td>
<td>(+)</td>
</tr>
<tr>
<td>(R_7)</td>
<td>(-) (+) (+) (-)</td>
<td>(+a_0 b_0) (+a_1 b_1)</td>
<td>(-)</td>
</tr>
<tr>
<td>(R_{11})</td>
<td>(+) (-) (+) (+)</td>
<td>(-a_0 b_0) (+a_1 b_1)</td>
<td>(+)</td>
</tr>
<tr>
<td>(R_{13})</td>
<td>(+) (+) (-) (-)</td>
<td>(+a_0 b_0) (+a_1 b_1)</td>
<td>(-)</td>
</tr>
<tr>
<td>(R_{14})</td>
<td>(+) (+) (+) (-)</td>
<td>(+a_0 b_0) (+a_1 b_1)</td>
<td>(+)</td>
</tr>
</tbody>
</table>

The even numbered Bell states are complements of each other \(B_0 = -B_2\) and the same is true for the odd numbered states \(B_1 = -B_3\). This suggests something about the square of the Bell operator and as expected, a higher dimensional version of the sparse invariants surfaces.

\[
(B B = (B)^2 = +1 - S_A S_B = [0 -0 -0 0 -0 -0] = I^-

(B)^4 = -1 + S_A S_B = [0+0+0+0+0+0] = I^+
\] (14)

An important question is, “Is the Bell operator singular?” The answer is yes, because \((B)^4\) does not exist [4], which means that once the Bell operator is applied, the combined states cannot be exited or escaped using a unitary operator. Applying the inverted operator \(-B\) evolves the states in the opposite direction \(B_{i-1} = B_i (-B)\).

How the Bell operator erases information can easily be demonstrated once the magic operator and magic states are defined. The four magic states \(^2\) \((M_0 => M_1 => M_2 => M_3 => M_0)\) are generated by the singular magic operator \(M = (S_A - S_B)\) using the iteration \(M_{i+1} = M_i M\). The magic states produce 90° out-of-phase sparse invariants compared to the Bell versions.

\(^1\) \(B_0 = -S_{00} + S_{11} = \Phi^-, B_1 = S_{01} + S_{10} = \Psi^+, B_2 = S_{10} - S_{01} = \Phi^-, B_3 = -S_{11} - S_{00} = \Psi^+\)

\(^2\) \(M_0 = S_{01} - S_{10}, M_1 = -S_{00} - S_{11}, M_2 = -S_{01} + S_{10}, M_3 = S_{00} + S_{11}\)
\[ M M = (M)^2 = +1 + S_A S_B = [-00 - 0 -0 -0 0 -0 00] = I^- \]
\[ (M)^4 = -1 - S_A S_B = [+00+ 0++0 0+0+] = I^+ \] (15)

It is possible to switch reversibly between the Bell and the magic states because \( M_3 = B_2 \) \((S_{01} + S_{10})\). An important relation for Bell and magic states is:

\[ \hat{B}_i M = M \hat{B}_i = 0, \]

which follows from the complete destructive interference of these state and operator spinors. Armed with this knowledge, one can usefully express the original entanglement equations as the sum of Bell and magic states.

\[ A B = (\pm a_0 \pm a_1)(\pm b_0 \pm b_1) = \pm a_0 b_0 \pm a_0 b_1 \pm a_1 b_0 \pm a_1 b_1 = B_j + M_i \] (16)

Some particular examples are:

\[ A_0 B_0 = (a_0 - a_1)(b_0 - b_1) = + a_0 b_0 - a_0 b_1 - a_1 b_0 + a_1 b_1 = B_3 + M_3 \]
\[ A_4 B_4 = (a_0 + a_1)(b_0 + b_1) = + a_0 b_0 + a_0 b_1 + a_1 b_0 + a_1 b_1 = B_1 + M_3 \] (17)

Therefore, independent of the starting state, half of the states are always multiplicatively erased when applying either the Bell or magic operators because \( B_i M = M_i B = 0 \). These results show that information is erased and these operators are irreversible, since a many-to-one mapping occurs due to erasure, as illustrated with the examples \( A_0 B_0 M \) and \( A_4 B_4 M \):

\[ A_0 B_0 B = B_0 + 0 \text{ and } A_0 B_0 M = 0 + M_0 \]
\[ A_4 B_4 B = B_2 + 0 \text{ and } A_4 B_4 M = 0 + M_0 \] (18)

A simple proof that \( B \) and \( M \) are singular can also be realized using the Cancellation Principle of Multiplication of multivectors which states: if \( X Y = X Z \) then \( Y = Z \) if and only if \( 1/X \) exists. The proof uses an example: if \( X = Y = B \) and \( Z = P_A P_B (-1) \), it can be shown that:

\[ B B = B \; (1 + S_A S_B) \text{ is always True but } \]
\[ Z = P_A P_B (-1) = -1 + S_A + S_B - S_A S_B = B - (1 + S_A S_B) \] (19)

The equality \( B = B - (1 + S_A S_B) \) can be true only if \( (1 + S_A S_B) = 0 \), which is always False even though the product \( B \; (1 + S_A S_B) = 0 \) is always True. This contradiction therefore means \( B \neq B - (1 + S_A S_B) \) because \( 1/B \) does not exist\(^*\) and \( B \) is singular. Similarly, \( M \) is singular. □

The Bell and magic states can also be used as singular operators\(^\ddagger\) to orient the states, because:

\[ B_i \; B_i = I^-, \; B_i \; B_{i+2} = I^+ \text{ while } B_i \; B_{i+1} = B_i \; B_{i+1} = \text{random states, and likewise for } M_i. \]

See Figure 8 for a graphical summary of the states, where \( P_{AB} = P_A P_B \). It is easy to understand that for three (or more) qubits, there are \((q-1)^2 = 4\) equivalent Bell operators of the form \((S_A \pm S_B \pm S_C)\) and the same number of out-of-phase sets of Bell states with exactly the same properties discussed here. This concludes the discussion of ebits and the Bell and magic states. The next topic is the new operators that are possible for two qubits.

---

\(^*\) Exhaustively searched the 43 million cases for solutions \( X \in Q_2 \) where \((S_A \pm S_B)(X) = 1\) and found none.

\(^\ddagger\) All \( B \) and \( M \) are singular because they respectively contain \( B \) and \( M \) as factors.
4.2. Conditional Operators CNOT and CSPIN

The only logic-like operator for one qubit is inversion due to phase spinning. The new operators possible for two qubits are the so-called conditional operators (similar to the familiar if-then-else clauses) because one qubit acts as a control qubit forming a conditional gating state for the operator action on the other data qubit. Three or more qubits are required before conventional logic operations can be performed using fully reversible logic gates such as the Toffoli and Fredkin gates.

The conditional form of inversion is called the control-not operator (CNOT) and the conditional spinor is called the control-spin operator (CSPIN). Both the CNOT and CSPIN operators are expressed as multivector operators that are applied using the geometric product. Conditional operators have the general behavior that if the state of a control qubit $A$ is in state $A_1$ then the operation is performed on data qubit $B$. Alternately if qubit $A$ is in state $A_0$ then the operation is not performed on qubit $B$. The CNOT operator performs a conditional inversion of the data qubit, while leaving the control qubit unchanged.

Conditional operators are conceptually tricky with regard to quantum computing for the following reasons. First, it is easy to assume, based on classical computing ideas, that in order to “know” the state of the control qubit, it must be measured, which is problematic, if measurement erases information. Second, therefore the conditionality must occur by applying specific operators only to specific states. This is also problematic since the states are thoroughly mixed via entanglement, and it is hard to separate out just the ones you want. Third, geometric products of multivectors are unconditional since each n-vector element is jointly affected by every n-vector in the operator. The results achieved so far for one qubit are due to the natural unconditional behavior of geometric products, spinors, and destructive interference.

An example of a conditional operator for one qubit is the reverse operator, denoted as $\hat{A}$. As the name suggests, this operator simply reverses the order of the vectors in an n-vector $A$, but

---

1 Control-spin is usually called a control-Hadamard gate in the literature.
2 Reverse is identical to Hermitian adjoint $\hat{A} = A^\dagger$ used in matrices. If $\hat{A} = A$ then $A$ is self-adjoint
this is not related to the concept of reversibility. If the vectors are then placed back in the standard vector order, then dependent on the overall grade of the particular n-vector, the coefficient will conditionally either remain the same or complement its orientation due to anticommutative operand swaps. The reverse of a multivector is the reverse of each graded element separately, where scalars and vectors are unaffected. Here are some examples.

\[
\text{reverse}(\pm 1) = \pm 1 \quad \text{and} \quad \text{reverse}(a) = a \\
\text{reverse}(a \ b) = b \ a = -a \ b \\
\text{reverse}(a \ b \ c) = c \ b \ a = -a \ b \ c \\
\text{reverse}(a \ b \ c \ d) = d \ c \ b \ a = +a \ b \ c \ d \\
\text{reverse}(a \ b \ c \ d \ e) = e \ d \ c \ b \ a = +a \ b \ c \ d \ e \\
\text{reverse}(a \ b \ c \ d \ e) = f \ e \ d \ c \ b \ a = -a \ b \ c \ d \ e \\
\]

Through use of the reverse operator and the operator \( A_0 = (a_0 - a_1) \), a single qubit \( A \) can be reversibly encoded into the even-grade plane to represent a complex number (\( A_0 A_0 = -1 \), \( A_1 A_0 = -S_A \), \( A_0 A_0 = S_A \)). The operator equivalent to the requisite complex conjugate can then be performed using the reverse operator to invert conditionally only the sign of the imaginary (or bivector) portion. This result is then converted back into the standard qubit states using the operator \( A_1 = (-a_0 + a_1) \). This sequence of steps \( A' = \text{reverse}(A \ A_0) \ A_1 \) conditionally inverts only the superposition states \( A_\alpha \) and topologically represents a reflection of the states off one of the axis, but cannot be realized by using only the unconditional geometric product. The main point of this discussion is that in general, writing conditional operators in a reversible linear representation is not straightforward and requires specialized state preparation and operators (e.g. conjugation) other than geometric products. In spite of this general restriction, it is possible to realize CNOT and CSPIN as multivector operators.

The earlier point regarding knowing the state of the control qubit is the inspiration behind the CNOT operator. As shown above for the complex number representation of a qubit, it is possible to encode a qubit in the even-grade plane using the operator \( A_0 = (a_0 - a_1) \). The classical states \( A_0, A_1 \) are mapped to \( \pm 1 \) respectively (an invariant) and the superposed states \( A_\alpha \) are mapped to \( \pm S_A \) (a random value). So the result of using any state as its own operator is like making a reversible encoding without breaking the symmetry of the qubit. This insight is the key to understanding that the control-not operator for control qubit \( A \) is \( CNOT_{AB} = A_0 \). Here are the results of entangling two qubits with the application of the CNOT operator.

\[
A_0 \ B \ CNOT_{AB} = (+1) \ B = +B \Rightarrow \text{leave data qubit} \\
A_1 \ B \ CNOT_{AB} = (-1) \ B = -B \Rightarrow \text{invert data qubit} \\
A_0 \ B \ CNOT_{AB} = (+S_A) \ B = S_A(+)B \Rightarrow \text{leave data qubit} \\
A_1 \ B \ CNOT_{AB} = (-S_A) \ B = S_A(-)B \Rightarrow \text{invert data qubit} \\
\]

As expected, the CNOT operator maps the control qubit to the other encoding, but the right multiplication of the operator causes the sign to become inverted due to the non-commutative operation \( B \ A_0 = -A_0 \ B \). The overall effect is to invert \( B \) depending on the state of \( A \). It is useful to think that this reversible operator reassigns the information in qubit \( A \) to the sign of qubit \( B \) (remember \( A_0 B_1 = A_1 B_0 \)). So qubit \( A \) now contains the state \(+1\), which means \( A \) was classically encoded and \(+S_A\) means \( A \) was encoded as a superposition. A control-not gate is intended to be defined only for classical control states, so the result containing the spinor \( S_A \).
is correct. The same analysis derives the operator when the roles are swapped for the data and control qubits. Another way to think of this is that $A_1$ and $B$ define a simultaneous constraint. This result is not exactly the conventional definition of the control-not operator since the encoding of the control qubit is modified. This can be remedied if another qubit $A'$ is initialized to the same state $A' = A$, then the result is that the new qubit $B$ includes a duplicate of the entangled information from $A$, and the qubit $A$ is left intact and untouched. The duplicate must be created in parallel since copying or cloning a qubit requires a measurement. This restriction is called the no-cloning theorem of quantum information.

$$A \ A' \ B \ \text{CNOT}_{A'B} = A \ (\mp \ B) = \mp A \ B$$

(22)

Since $(S_A)^2 = (\text{spinor})^2 = \text{NOT}$ the inspiration occurred to solve for $(\text{CSPIN})^2 = \text{CNOT}$, and the result is $	ext{CSPIN} = \sqrt{\text{CNOT}} = -I + A_0$ (and its other root, and inverse of $+I + A_1$). This operator has the same concurrent structure as the Pauli spin operator, except with the concurrent operators being the inversion and reversible encoding. Since $	ext{CSPIN} = \sqrt{-I}$ it indicates a 45 degree rotation. Interestingly, the Bell operators have this exact same structure where $(B)^2 = -I$, and $B = (B)^2 + B = I + B$ and this structural similarity of equations is most likely a meaningful coincidence. The results of the CSPIN operator in Eq. (23) and Table 9 are interesting because they show the need for a mixed-grade multivector to encode the phase information.

$$A_0 \ B_0 \ \text{CSPIN}_{A'B} = B_0 - A_0 \ B_0 = (b_0 - b_1) + a_0 \ b_0 + a_1 \ b_1 - a_1 \ b_1$$

$$A_0 \ B_0 \ \text{CSPIN}_{A'B} = S_A \ B_0 + A_0 \ B_0 = a_0 \ a_1 \ (b_0 - b_1) + a_0 \ b_0 - a_0 \ b_1 + a_1 \ b_0 - a_1 \ b_1$$

(23)

For classical states of the control qubit $A$, Table 9 shows that the overall multivector orientation inverts depending on the control qubit state. The superposition states are also encoded, yet of the 16 possible rows only 6 rows are valid at once. The valid rows indicate what the valid states are and represent a simultaneous constraint system where the operators conditionally change the overall row states that are non-zero. This is clearly evident by the conditional validity of row-states $R_5$, $R_6$, $R_9$ and $R_{10}$ in Table 9.

<table>
<thead>
<tr>
<th>Row, $k$</th>
<th>Combinations</th>
<th>Active States $A \ B \ \text{CSPIN}<em>{A'B} = \mp A \ B + B</em>{01}$</th>
<th>$A_0 B (A_0\mp 1)$</th>
<th>$A_1 B (A_0\mp 1)$</th>
<th>$A_2 B (A_0+1)$</th>
<th>$A_3 B (A_0+1)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$R_1$</td>
<td>$-$ $-$ $-$ $+$</td>
<td>$A_1 &amp; B_1$</td>
<td>$+$ $-$ $+$ $-$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$R_2$</td>
<td>$-$ $-$ $+$ $-$</td>
<td>$A_0 &amp; B_0$</td>
<td>$-$ $+$ $-$ $+$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$R_5$</td>
<td>$-$ $+$ $+$ $-$</td>
<td>$A_1 &amp; B_1$</td>
<td>$0$ $0$ $-$ $+$</td>
<td>$= b_0$ $+$ $-$</td>
<td>$= b_1$</td>
<td></td>
</tr>
<tr>
<td>$R_6$</td>
<td>$-$ $+$ $-$ $+$</td>
<td>$A_1 &amp; B_1$</td>
<td>$0$ $0$ $+$ $-$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$R_{10}$</td>
<td>$+$ $-$ $-$ $+$</td>
<td>$A_0 &amp; B_0$</td>
<td>$-$ $+$ $- b_0$</td>
<td></td>
<td>$0$ $0$</td>
<td></td>
</tr>
<tr>
<td>$R_{13}$</td>
<td>$+$ $+$ $-$ $+$</td>
<td>$A_0 &amp; B_0$</td>
<td>$+$ $-$ $-$ $+$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$R_{14}$</td>
<td>$+$ $+$ $+$ $-$</td>
<td>$A_1 &amp; B_1$</td>
<td>$-$ $+$ $-$ $+$</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

This concludes the new operators for $Q_2$. 


5. Toffoli Operator is Concurrent CNOT

The same process for the control-not gate can be expanded to \(Q_3\) in order to include two control qubits \(A, B\) and a data qubit \(D\). The resulting control-control-not gate is called the Toffoli operator and only inverts qubit \(D\) when the control qubits are both active (denoted by the subscript 1) in states \(A_1\) and \(B_1\). The individual cases of single control-nots are first expressed to correctly account for the anticommutative operand swaps. The control qubits are indicated by the small subscript \(c\), since it is not always the first one listed in an expression.

\[
\begin{align*}
A B D \text{ CNOT}_B D &= A B D (B_0) = A B_1 B_1 D = \pm A D \quad \text{(one operand swap)} \\
A_c B D \text{ CNOT}_AD &= A_c B D (A_1) = A_c A_1 B D = \pm B D \quad \text{(two operand swaps)}
\end{align*}
\]

Now the Toffoli Operator is

\[
\text{TOF}_{ABD} = \text{CNOT}_AD + \text{CNOT}_BD = A_1 + B_0 = (-a_0 + a_1 + b_0 - b_1)
\]

and is reversible because \((\text{TOF})^2 = +1\). This simple grade-1 multivector operator and grade-2 multivector outcome is a direct result of applying the concurrency interpretation of addition as discovered for the Bell operator. Here is the general Toffoli gate formula:

\[
A_c B_c D (\text{TOF}_{ABD}) = A_c B_c D (A_1 + B_0) = \pm B D \pm A D 
\]  

(24)

An particular case of Eq. (24) is now required in order to compute the valid rows in Table 10:

\[
A_0 B_0 D_0 (\text{TOF}_{ABD}) = + a_0 d_0 - a_0 d_1 - a_1 d_0 + a_1 d_1 + b_0 d_0 - b_0 d_1 - b_1 d_0 + b_1 d_1 = [00000+0-00000+00-000000+0-000000+00000+0-000000+00000] 
\]

(25)

Table 10. Valid row states for \(A_0 B_0 D_0 (\text{TOF}_{ABD})\) in \(Q_3\)

<table>
<thead>
<tr>
<th>Row (k)</th>
<th>State Combinations</th>
<th>Active States</th>
<th>(A_0 B_0 D_0 (\text{TOF}_{ABD}))</th>
</tr>
</thead>
<tbody>
<tr>
<td>(R_{21})</td>
<td>(- - + - + + -)</td>
<td>(A_1 B_1 &amp; D_1)</td>
<td>- Inverted</td>
</tr>
<tr>
<td>(R_{22})</td>
<td>(- + - + + -)</td>
<td>(A_1 B_1 &amp; D_0)</td>
<td>+</td>
</tr>
<tr>
<td>(R_{41})</td>
<td>(+ - + - - +)</td>
<td>(A_0 B_0 &amp; D_1)</td>
<td>+ Identity</td>
</tr>
<tr>
<td>(R_{42})</td>
<td>(+ - - - + -)</td>
<td>(A_0 B_0 &amp; D_0)</td>
<td>- Mixed states</td>
</tr>
<tr>
<td>8 rows</td>
<td>(A_{\text{classical}}) (B_{\text{superpose}}) (D_{\text{classical}})</td>
<td>(A_1, B_1, &amp; D_0)</td>
<td>±</td>
</tr>
<tr>
<td>44 rows</td>
<td>All conditions not listed above</td>
<td>none</td>
<td>0 Invalid</td>
</tr>
</tbody>
</table>

Rows 21-22 in Table 10 represent the valid states where both control lines are active high and the output orientation is inverted compared to qubit \(D\). Rows 41-42 represent the valid states when no inversion occurs, so the output orientation matches qubit \(D\). Since the Toffoli gate \(\text{TOF}_{ABD} = (-a_0 + a_1 + b_0 - b_1)\), it is clear why three qubits in \(Q_3\) are necessary to express this operator. There are four variants of this operator, \(A_0+B_0, A_1+B_0, A_1+B_0, \text{ and } A_1+B_1\), depending on the desired Boolean condition.

Notice that no other row states are valid when both controls have classical states! This is important because, due to the overall symmetry in geometric algebra, designing arbitrary multiplicative operators is difficult, so in essence operators are discovered, not designed. This problem is akin to building a ship in a bottle, where the quantum state is analogous to a very high-dimensional bottle and only tools (or operators) that fit through the neck of the bottle (combinations of single qubit operators) are allowed. It is possible to design an
arbitrary state because the row states are linearly independent (given any vector notation can uniquely convert to the algebraic notation and vice versa). Some states can only be created via addition rather than with a multiplicative operator starting from a valid entangled qubit state.

6. Conclusions

The wealth of quantum computing concepts described here, using only addition and geometric products, is possible because geometric algebra naturally and implicitly captures the topological informational distinctions and constraints needed to represent qubits, ebits and familiar operators. This is the only possible interpretation of the co-occurrence of two vectors. Due to the power of geometric algebra to represent classical mechanics, gravitational contraction and quantum mechanics, it is called “a unified language for physics and engineering” [5]. This work extends that domain to include quantum information and quantum computation with straightforward, well-developed [4] and – most importantly – easily interpreted mathematics. This work presents a qubit algebra and as well demonstrates a linearly independent, dual, vector notation that is useful because it combines the topologically smallest elements in the algebra.

It is interesting to see how unfamiliar but transparently meaningful algebraic rules emerge directly from the choice of symmetric binary values +1 and –1 and the mapping of co-occurrence and co-exclusion to addition and the geometric product, i.e. $a b = -b a$ and $a a = 1$. This symmetry then impacts the symmetry of the addition and multiplication operators, i.e. $1/a = a$, $2a = a + a = -a = a/2$ and enables sparse invariants. This symmetry is reinforced because qubits are the sum of two vectors, which results in many counts being a power of 2. As a result, the additive and multiplicative inverses become interchangeable as $A_0 = -A_1 = 1/A_1$, but also sequential and concurrency ideas herewith intersect, e.g. $R_k R_k = R_k + R_k = P_k$. One should remember that the mathematics describing quantum mechanics is algebraically closed, and so is equivalent to bouncing a light beam around inside a hollow mirrored sphere.

Quantum computing works because it relies on the intrinsically high-dimensional infrastructure of the quantum universe. John Wheeler’s paper “It from Bit” [13] stipulates that everything classical, including energy, matter, spacetime and even empty space, emerges from this bit soup (also called quantum ether or quantum foam) because the universe started as a “bit bang” [6,12]. Our geometric algebra approach algebraically and consistently describes topological quantum information forms as a massless high-dimensional topology and true concurrency without focusing on how it is projected into any of the classical properties of space, time or energy. This approach is consistent with extant quantum gravity theories treating the information mechanics of black holes (or bit buckets) [14].

It is possible to make better decisions, to be smarter, with high-dimensional spaces [15] because more states can participate simultaneously in a decision, due to a higher locality metric and true concurrency. Quantum metrics and phenomena are not possible in computation restricted to classical spacetime. Spacetime itself limits the computational density by segregating [16] the required information locality and concurrency. This alone should motivate engineers and programmers to want to understand quantum computing: because it allows computers to cheat by computing outside the limiting spacetime box that occurs when representing bits classically. Because of the unusual and counterintuitive nature of quantum information, encouraging engineers and programmers to ascend the quantum computing learning curve will lead to an appreciation of the fundamental role of information in the quantum computing universe and might lead to general purpose quantum computers.
Acknowledgements

Special thanks go to my UTD Ph.D. committee chairmen and members. Also, I would like to thank Katrina Riehl, who helped outline the sequence of quantum computing concepts that would be useful (and also avoided) for a novice computer professional.

References