Electronic Design

RUSSELL J. HOPPENSTEIN, Sr. Principal Application Engineer,
Qorvo Inc.

Fundamentals of Reversible Logic

A new type of logic supports high-performance computing and quantum computers.

In basic engineering classes, students learn the fundamentals of logic circuits. These logic circuits are the foundation of computing. There's an inherent limitation in the basic logic circuits, whereby each operation potentially wastes a smidgeon of energy. For most applications, the wasted energy isn't consequential, but for high-performance computers (HPCs) that execute billions upon billions of calculations, this inefficiency is undesirable. Thus, it's induced a new type of logic: reversible logic.

Reversible Logic

In conventional logic, certain operations erase information. Per Landauer's Principle, erasing 1 bit of logic dissipates energy corresponding to Equation 1.

$$E \geq k \cdot T \cdot \ln(2) \quad (1)$$

With reversible logic, the inputs are maintained and the results can be reversed. There are two primary postulates for reverse logic.

- Each output maps from a unique input.
- Each input can be reconstructed from the output.

You can visualize a truth table where each input variable correlates to a unique output. The invertibility requirement can be expressed mathematically in matrix form in Equation 2.

$$X^{-1} \cdot X = I \quad (2)$$

It's a fancy way of saying that when you pass the output values through the same matrix calculations, you will get back to the inputs.

Simple Logic

Let's take a simple NOT gate. This is inherently reversible as each input yields a unique output. In reversible logic, a CNOT (Controlled NOT) gate is used to preserve reversibility while allowing for conditional operations. The CNOT gate has a control bit and the operating bit. The operation is as follows:

- If the control bit is 0, do nothing, and pass inputs to the outputs.
- If the control bit is 1, operate on the operating bit (i.e., execute the NOT operation).

Table 1 shows the modified truth table.
Mathematically this is expressed in Equation 3.

$$(a,b) \rightarrow (a, a \text{ XOR } b) \quad (3)$$

A quick check of the truth table illustrates that each unique input has a unique output. If you feed the output

Table 1: CNOT Gate Truth Table

Control Bit	Target Bit	Control Out	Target Out
0	0	0	0
0	1	0	1
1	0	1	1
1	1	1	0

Table 2: AND Gate Truth Table

Bit 1	Bit 2	Output
0	0	0
0	1	0
1	0	0
1	1	1

through the circuit, you return to the original input values. This gate is truly reversible.

More Complicated Logic

Now let's look at some logic that's a bit more complicated (pun intended). Take a traditional AND gate per the truth table in *Table 2*.

This one isn't intrinsically reversible. Two inputs result in a single output. Knowing just the output, it isn't possible to reconstruct the input. The inputs are ultimately "discarded" once the operation is complete.

To make the AND gate reversible, the input parameters must be passed to the output so that there will be three outputs. The input also needs three connections; therefore, the third connection is an ancilla bit set to 0. To be fully reversible, the input sequence must map uniquely to an output sequence, and that sequence must map back to the original sequence. But how? The Toffoli gate is the solution.

Toffoli Gate

The Toffoli gate is a three-input, three-output reversible gate designated as a CCNOT (Controlled, Controlled NOT) gate. It's the general building block for reversible logic. The CCNOT is defined in Equation 4.

$$(a, b, c) \rightarrow (a, b, c \text{ XOR } (a \text{ AND } b)) \quad (4)$$

The inputs *a* and *b* are mapped directly to the output. The target value is related to the inputs so that it can be reversed. *Table 3* shows the truth table for the CCNOT gate.

Though the Toffoli gate in this case may look like a standard AND gate, it's more pertinent to think about it as a controlled inversion of the last bit. In *Table 3*, most of the inputs map directly to the same output, so it's straightforward to see how reversing the structure will yield the original results. The tricky one is the last element. Equation 5 illustrates the reverse calculation for that last element.

$$(1, 1, 0) \rightarrow (1, 1, 1) \rightarrow (1, 1, 1 \text{ XOR } (1 \text{ AND } 1) = (1, 1, 0) \quad (5)$$

So, the last output does indeed reverse to its original form. All of the data is maintained and nothing wasted. The improvement of efficiency is weighed against the penalty of needing more connections than before. For example, the CCNOT requires three inputs and three outputs as opposed to the traditional AND gate that only needs two and one.

Does the Efficiency Improvement Really Help?

So, why do I care about this efficiency improvement? Does it really help me? For most tasks, likely no. Traditional logic will do just fine. But when the calculations shift up to the exascale (a million trillion operations) for HPC applications, the efficiency gains may be attractive after other inefficiencies are optimized. Once the final answer from a large calculation is found, all of the intermediate steps are reversed to preserve the input and not waste the information (as heat).

The other emerging area for reversible circuits is quantum computing. In this case, reversible logic is mandatory and not a luxury. The nature of quantum qubits is that they must remain ambiguous or entangled (that spooky action at a distance per Einstein's quote) as the computations unfold. Only at the last final operation does the result collapse to a single value. Intermediate states evolve coherently and cannot be erased without measurement, as any measurement will collapse the quantum state.

Conclusion

Reversible logic will not replace traditional logic. For many standard operations, the classical operations work fine and the heat dissipation from lost bits is thermally manageable. Emerging HPC systems and quantum computers will require reversible logic. These systems will be the foundation for the next evolution in artificial intelligence and complex computational simulations.

Russell Hoppenstein, an RF application engineer at Qorvo, supports high-speed RF and timing devices.

Table 3: Reversible CCNOT Gate Truth Table

Bit 1	Bit 2	Ancilla	Bit 1 Out	Bit 2 Out	Target Out
0	0	0	0	0	0
0	1	0	0	1	0
1	0	0	1	0	0
1	1	0	1	1	1