Implementation of High Speed Low Power Combinational and Sequential Circuits using Reversible logic

1P.Rajasekhar Reddy, 2S.Raghavendra Swami, 3S.Ravi Kumar

1PG Scholar, Digital Electronics and Communication Systems, P.V.K.K. Institute of technology, AP, India
2Assistant Professor, Electronics and Communication Engineering, P.V.K.K. Institute of technology, AP, India
3Assistant Professor, Electronics and Communication Engineering, P.V.K.K. Institute of technology, AP, India (rajasekharreddy432@gmail.com)

Abstract— Reversible logic has presented itself as a prominent technology which plays an imperative role in Quantum Computing. Quantum computing devices theoretically operate at ultra high speed and consume infinitesimally less power. Research done in this paper aims to utilize the idea of reversible logic to break the conventional speed-power trade-off, thereby getting a step closer to realise Quantum computing devices. To authenticate this research, various combinational and sequential circuits are implemented such as a 4-bit Ripple-carry Adder, (8- bit X 8-bit) Wallace Tree Multiplier, and the Control Unit of an 8-bit GCD processor using Reversible gates. The power and speed parameters for the circuits have been indicated, and compared with their conventional non-reversible counterparts. The comparative statistical study proves that circuits employing Reversible logic thus are faster and power efficient. The designs presented in this paper were simulated using Xilinx 9.2 software.

Key Terms— Reversible logic, Quantum Computing, highspeed, less power, speed-power trade-off, Ripple carry adder, Wallace tree multiplier, GCD processor, non-reversible counterparts.

I. INTRODUCTION

Reversible logic is widely used in low power VLSI. Reversible circuits are capable of back-computation and education in dissipated power, as there is no loss of information [1]. Basic reversible gates are employed to achieve the required functionality of a reversible circuit. The uniqueness of reversible logic is that, there is no loss of information since there is one-to-one correspondence between inputs and outputs. This enables the system to run backwards and while doing so, any intermediate design stage can be thoroughly examined. The fan-out of each block in the circuit has to be one. This research paper focuses on implementation of reversible logic circuits in which main aim is to optimize speed of the design. A Reversible adder is designed using basic reversible gates. Using this adder, an 8-bit reversible ripple-carry adder is devised and then compared with the conventional 8-bit adder in terms of speed, critical paths, hardware used. Then using the same reversible adder, a Wallace tree multiplier has been implemented, and compared with the conventional Wallace tree multiplier. With the known fact that sequential circuits are the heart of digital designing, the design for the control unit of a reversible GCD processor has been proposed using Reversible logic gates.

II. REVERSIBLE LOGIC

Boolean logic is said to be reversible if the set of inputs mapped have an equal number of outputs mapped i.e. they have one-to-one correspondence. This is realized employing reversible gates in the designs. Any circuit having only reversible gates is capable of dissipating no power.

Goals of Reversible Logics:

A. Quantum Cost: Quantum cost of a circuit is the measure of implementation cost of quantum circuits. More precisely, quantum cost is defined as the number of elementary quantum operations needed to realize a gate.

B. Speed of Computation: The time delay of the circuits should be as low as possible as there are numerous computations that have to be done in a system involving a quantum processor; hence speed of computation is a very important parameter while examining such systems.

C. Garbage Outputs: Garbage outputs are those output signals which do not contribute in driving further blocks in the design. These outputs become redundant as they are not required for computation at a later stage. The garbage outputs make the system slower; hence for better efficiency it is necessary to minimize the number of garbage outputs.

D. Feedback: Looping is strictly prohibited when designing reversible circuits.

E. Fan-out: The output of a certain block in the design can only drive at most one block in the design. Hence it can be said that the Fan-out is restricted to 1.
III. MAJOR SEVERAL REVERSIBLE LOGIC

3.1 Feynman Gate
It is a 2*2 Feynman gate [13]. The input vector is I (A, B) and the output vector is O(P, Q). The outputs are defined by P=A, Q=AÅB. Quantum cost of a Feynman gate is 1. Figure 1 shows a 2*2 Feynman gate.

![Figure 1: Feynman gate](image1)

3.2 Double Feynman Gate (F2G)
It is a 3*3 Double Feynman gate [14]. The input vector is I (A, B, C) and the output vector is O (P, Q, R). The outputs are defined by P=A, Q=AÅB, R=AÅC. Quantum cost of double Feynman gate is 2. Figure 2 shows a 3*3 Double Feynman gate.

![Figure 2: Double Feynman gate](image2)

3.3 Toffoli Gate
It is a 3*3 Toffoli gate [6]. The input vector is I (A, B, C) and the output vector is O (P, Q, R). The output is defined by P=A, Q=B, R=ABÅC. Quantum cost of a Toffoli gate is 5. Figure 3 shows a 3*3 Toffoli gate.

![Figure 3: Toffoli gate](image3)

3.4 Fredkin Gate
It is a 3*3 Fredkin gate [7]. The input vector is I (A, B, C) and the output vector is O (P, Q, R). The output is defined by P=A, Q= AÅBÅC and R= AÅCÅB. Quantum cost of a Fredkin gate is 5. Figure 4 shows a 3*3 Fredkin gate.

![Figure 4: Fredkin gate](image4)

3.5 Peres Gate
It is a 3*3 Peres gate [15]. The input vector is I (A, B, C) and the output vector is O (P, Q, R). The output is defined by P=A, Q=AÅB and R=ABÅC. Quantum cost of a Peres gate is 4. Figure 5 shows a 3*3 Peres gate.

![Figure 5: Peres gate](image5)

3.6 Double Peres gate
It is a 4*4 Double Peres Gate [16]. The input vector is I (A, B, C, D) and the output vector is O (P, Q, R, S). The output is defined by P=A, Q=AÅB, R=AÅBÅD and S=(AÅB)DÅABÅC. Figure 6 shows a 4*4 Double Peres gate.

![Figure 6: DPG gate](image6)

IV. REVERSIBLE 4-BIT FULL ADDER
The gate used in implementing a reversible ripple-carry full adder is the TSG gate [4]. The TSG gate functions like a full adder. A reversible ripple-carry adder is faster than the non-reversible adder, since the computation of carry in a reversible adder does not require the computation of previous stage carry (as indicated in the critical paths). When previous stage carry is being forwarded in the reversible adder, the computation of previous stage carry and computation regarding sum is done simultaneously whereas in an irreversible adder the next stage carry cannot start any computation till previous stage carry is fully generated. The critical paths of 4bit reversible and irreversible ripple-carry adders are as shown in fig.6 and fig.7.

![Fig. 6: Critical Path of 4-bit reversible adder](image7)

![Fig. 7: Critical Path of 4 bit irreversible adder](image8)
V. WALLACE TREE MULTIPLIER

A Wallace tree is an efficient hardwired implementation of a digital circuit that multiplies two integers [5]. The Wallace tree has three steps:

1. Every bit of the multiplicand is multiplied (i.e. AND) by every bit of multiplier, thus yielding \( n^2 \) results (for \( n \times n \) multiplication). Depending on position of the multiplied bits, the wires carry different weights, i.e. weight of bit carrying result of a 6 is 65.

2. The number of partial products is reduced to 2 by layers of full and half adders.

3. The wires are grouped in two numbers, and added using a conventional adder. The circuit diagram of Wallace tree multiplier using reversible gates is shown in fig. 8.

VI. DESIGN OF CONTROL UNIT FOR GCD PROCESSOR

To illustrate the classical and reversible approaches to the Sequential Control Unit Design, reversible logic is employed for a special purpose processor that computes the GCD of two numbers. This GCD processor incorporates standard Euclid’s Algorithm involving Subtract-Compare-Swap operation of two numbers. The basic principle is to subtract smaller of the two numbers repeatedly from the other number until we get the number that divides another

A. Control Unit

Control unit of GCD processor generates the control signals to manipulate the operations in Data-path. Fig. 9: Block diagram of GCD Control Unit.

B.. Block Diagram Description:

1) Flip-flop Module:

The control unit for GCD processor requires two Flipflops as binary state encoding is used for FSM. In this design reversible edge-triggered D Flip-flop is employed for state transitions [7]. Two D-latches are connected in Master-Slave mode to act as an edge-triggered D Flip-flop. Reversible D-latch is designed using Feynman and Fredkin gates. RTL schematic of reversible D flip-flop obtained is shown in fig. 10.

Fig. 10: RTL schematic of Reversible D flip-flop

2) Regeneration Module

To avoid multiple fan-out condition in the design, it is necessary to duplicate signals used for computation of output and next state. The duplication of input signals is achieved using Feynman gates.

3) Output Module

The computation of the outputs and Next-state signals is done using reversible Fredkin gates. The functioning of output signals is driven by the algorithm.

C. Final RTL schematic:

The complete RTL schematic of GCD control unit is shown in fig. 11.

Fig 11: RTL schematic Diagram of GCD Control Unit

VII. CONCLUSION

In this paper, it can be seen that the performance of digital circuits can be enhanced using reversible gates and have compared 8-bit ripple carry reversible adder with an irreversible adder in terms of speed and power; thereby concluding that reversible designs are faster and power efficient. Furthermore, this concept is extended to
combinational circuits such as a Wallace tree multiplier using reversible gates, which were simulated and respective results validate prior inferences. Then a reversible sequential control unit of a GCD processor was designed. Thus, all the designs implemented were compared with their irreversible counterparts, and the speed and power parameters for the reversible designs were observed to have improved significantly.

REFERENCES


