Panihati Mahavidyalaya
Reversible Multiply Accumulate Unit

Done by

Subhrajit Das

Ritabh Rudra

Biswajit Kumar Pal

CMSADSE06P: Project

Guided by Prof. Biswanath Sen

August, 2021

Index

Acknowledgement 1

Abstract 2

Introduction 3

Multiply Accumulate Unit 5

Data Representation in MAC 6

Reversible Computation 8

Types of Reversible Computation 9

Reversible Logic Gates: 9

I. Feynman Gate 9

II. Toffoli Gate 10

III. Fredkin Gate 11

IV. Peres Gate 12

V. BJN Gate 13

Reversible Gate Architecture of Toffoli Gate 14

Reversible Gate Architecture of Fredkin Gate 15

Reversible Multiply Accumulate Unit 16

Booth's Multiplication 16

Adder/Subtractor using Reversible Gates 17

Reversible 4 to 1 MUX 18

Reversible D Flip-Flop 19

Reversible Information Shifter 20

Reversible Booth's Multiplier 22

Result 23

Conclusion **26**

References **27**

List of Figures

1 Basic block diagram of MAC 5

2 Feynman Gate 9

3 Toffoli Gate 10

4 Fredkin Gate 11

5 Peres Gate 12

6 BJN Gate 13

7 Diagram of Toffoli Gate using CMOS 15

8 Diagram of Fredkin Gate using CMOS 16

9 Flowchart of Booth's Algorithm 17

10 Proposed ADDER/SUBTRACTOR using Reversible Gates 18

11 Reversible 4 to 1 Mux 19

12 Reversible Realization of D Flip-Flop 20

13 Universal Shifter implemented using Reversible MUX 20

14 Proposed Booths Hardware Reversible Module 22

15 Flowchart of Booth's Multiplication with Reversible Data Loading and Cleaning 24

16 Output of Flowchart implementation in a Java Program 24

17 Output of Flowchart implementation in a Java Program 25

List of Tables

1 Truth Table of Feynman Gate 10

2 Truth Table of Toffoli Gate 11

3 Truth Table of Fredkin Gate 12

4 Truth Table of Peres Gate 13

5 Truth Table of BJN Gate 14

6 Quantum cost and Garbage outputs of Reversible Modules 23

Acknowledgement

The Reversible Multiply Accumulate Unit can be used in embedded systems as the core component of Digital Signal Processing which reduces the power consumption.
In this paper we have proposed the different reversible modules towards our ultimate goal, i.e., development of Reversible Multiply Accumulate Unit. The intermediate reversible modules(Adder/Shifter, Information Shifter) also consumes less power.

We are grateful to our project guide, for his guidance, encouragement, valuable suggestions, innovative ideas and supervision throughout our work. We express our earnest and sincere appreciation to him as without him it would have been difficult for us to complete our work.

We also express our sincere thanks to all teachers of the department for their kind co operation and suggestion throughout the development of the project.

We would like to express our gratitude to the library staffs and laboratory staffs for providing us with a congenial working environment.

Abstract

In the state of the art,computing system design is irreversible. Reversible computing system is of interest because it is associated with very low power design. Nowadays approximately three billion embedded CPUs are sold each year with smaller CPUs dominating by quantity and aggregate dollar amount.A large section of embedded device use digital signal processing as its primary application.This Project proposed a reversible design of Multiply-Accumulate(MAC)Unit. As the MAC unit is a primary unit of signal processing system,its reversible processing will help us by saving large amount of energy.This project also proposed a reversible design of Adder/Subtractor and information Shifter which will be helpful for addition/subtraction and shifting information at a very low power energy.

Introduction

Multiplier is an important basic building block in designing of systems using digital signal processing (DSP) and in other applications such as microprocessors, micro-controller and other data processing unit.It control the execution time of overall system. Reversible logic gates establishes reversible logic circuits and their major application can be seen in quantum computing. Each quantum logic gate performs an elementary unitary operation on one, two or more two-state quantum system called qubits. In the design of reversible circuits, fan-out and loops are not permitted.From the point of view of reversible circuit design, there are three parameters for determining the complexity and performance of circuits:

Quantum cost (QC): The number of 1x1 or 2x2 or 3x3 reversible gates which are used in circuit.

Constant inputs (CIs): The number inputs which are used as control inputs and are connected to logical zero or one.

Garbage outputs (GOs): The number of dummy (unused) outputs which are made to appear in order to make the circuit reversible.

Gate count: The number of 1x1 or 2x2 or 3x3 reversible gates which are used in the design.

A reversible circuit designer always tries to develop on these four parameters. All Digital Signal Processing (DSP) algorithms extensively use Multiply-Accumulate (MAC) operation for high performance digital processing system. This operation eases the computation of convolution which is needed in filters, Fourier Transforms, etc. A MAC unit comprises of a multiplier, an adder and an accumulator. The multiplier multiplies the inputs and gives the result to the adder, which adds the multiplier result to the previously accumulated result. One of the strongest motivations for the study of reversible computing comes from the desire to reduce heat dissipation in computing machinery, and thus achieve higher density and speed. Briefly, while the microscopic laws of physics are presumed to be strictly reversible, abstract computing is usually thought of as an irreversible process, since it may involve the evaluation of many-to-one functions. Thus, as one proceeds down from an abstract computing task to a formal realization by means of a digital network and finally to an implementation in a physical system, at some level of this modeling hierarchy there must take place the transition from the irreversibility of the given computing process to the reversibility of the physical laws. In the customary approach, this transition occurs at a very low level and is hidden–so to speak–in the "physics" of the individual digital gate;* as a consequence of this approach, the details of the work-to-heat conversion process are put beyond the reach of the conceptual model of computation that is used. An important advantage of our approach is that any operations (such as the clearing of a register) that in conventional logic lead to the destruction of macroscopic information, and thus entail energy dissipation, here can be planned at the whole-circuit level rather than at the gate level, and most of the time can be replaced by an information- lossless variant. As a consequence, it appears possible to design circuits whose internal power dissipation, under ideal physical circumstances, is zero. The power dissipation that would arise at the interface between such circuits and the outside world would be at most proportional to the number of input/output lines, rather than to the number of logic gates.

Contribution: In this paper, a Reversible Multiply Accumulate Unit (MAC) is proposed. The reversible multiplier is implemented by the combination of reversible adders, reversible subtractors, reversible multiplexer,reversible information shifter, reversible D Flip-Flop, Feynman gate, Toffoli Gate, Fredkin Gate, Peres Gate and BJN Gate. The reversible adder is used as the adder and the reversible accumulator is designed using the reversible shift register. A reversible MAC unit is also built and compared with other possible implementations unit in terms of gate count, quantum cost, constant input and garbage output of the circuit.

Multiply Accumulate Unit

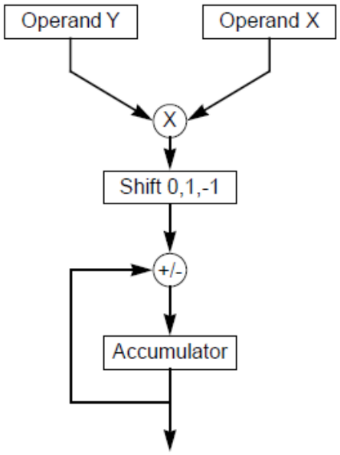


Figure 1

Basic block diagram of MAC

Digital signal processing(DSP) is considered as one of the important technologies in rapidly growing application areas such as audio and video processing, wireless communications,and industrial control. Digital signal processing is the mathematical manipulation of digitally represented signals.DSP processors perform this DSP operations.DSP applications require [Malik and Dhall (2012)] critical operations which usually involve many multiplications and accumulations. Hence, high throughput multiplier accumulator (MAC) is always a key element to achieve a high-performance digital signal processing application for real time signal processing applications.

Multiply-accumulate operation is one of the basic arithmetic operations extensively used in modern digital signal processing (DSP).

The MAC [Das, Kanhe, and Talwekar (2013)] unit provides high-speed multiplication, multiplication with cumulative addition, multiplication with cumulative subtraction, saturation, and clear-to-zero functions. Hence, the main goal is to analyze various pipe-lined MAC architectures and circuit and the design techniques which provides high throughput.

The Multiply Accumulate Unit(MAC) supports for a limited set of digital signal processing (DSP) hardware operations used in embedded code involving the integer multiply instructions.
The MAC unit works in three related fields:

- Signed and unsigned integer multiplications
- Multiply and Accumulate operations involving signed,unsigned.
- Miscellaneous register operations

The MAC operation modifies an accumulator a:

a <– a + (b * c)

It involves two stages

- Multiply
- Forward result for addition/accumulate

It is an add-on of the basic multiplier unit found on most microprocessors which performs operations based on signal processing algorithms in an acceptable number of cycles, given the application constraints.

Data Representation in MAC

The MAC unit supports two basic operand types:

- Two's complement signed integer: In this format, an N-bit operand represents a number within the range

  $$-2^{(N-1)} < operand < 2^{(N-1)} - 1. \tag{1}$$

  The binary point is to the right of the least significant bit

- Two's complement unsigned integer: In this format, an N-bit operand represents a number within the range

  $$0 < operand < 2^N - 1 \tag{2}$$

  The binary point is to the right of the least significant bit.

Reversible Computation

[Frank (2005)] Reversible computing is motivated by the von Neumann-Landauer (VNL) principle, a theorem of modern physics telling us that ordinary irreversible logic operations (which destructively overwrite previous outputs) incur a fundamental minimum energy cost. Such operations typically dissipate roughly the logic signal energy, itself irreducible due to thermal noise. This fact threatens to end improvements in practical computer performance within the next few decades. However, computers based mainly on reversible logic operations can reuse a fraction of the signal energy that theoretically can approach arbitrarily near to 100 percent as the quality of the hardware is improved, reopening the door to arbitrarily high computer performance at a given level of power dissipation. In the 32 years since the theoretical possibility of this approach was first shown by Bennett, our understanding of how to design and engineer practical machines based on reversible logic has improved dramatically, but a number of significant research challenges remain, e.g., (1) the development of fast and cheap switching devices with adiabatic energy coefficients well below those of transistors, (2) and of clocking systems that are themselves of very high reversible quality; and (3) the design of highly-optimized reversible logic circuits and algorithms. Finally, the field faces an uphill social battle in overcoming the enormous inertia of the established semiconductor industry, with its extreme resistance to revolutionary change. A more evolutionary strategy that aims to introduce reversible computing concepts only very gradually might well turn out to be more successful. This talk explains these basic issues, to set the stage for the rest of the workshop, which aims to address them in more detail.

Types of Reversible Computation

There are two types of Reversible Computation-

- **Physically Reversible:** For this type of computation,device running backwards for proceed the process.This case is possible upto 90 percent time.But, no device is fully physically reversible in real life scenarios.
- **Logically Reversible:** For this type of computation, Input and Output can't be reversible in physical science but we can try to make our processing of execution in reversible mode. In real life scenarios, this type is mostly applied in device.

Reversible Logic Gates:

I. Feynman Gate

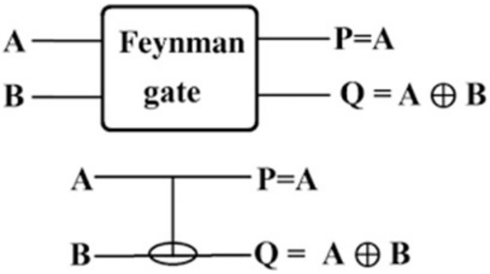


Figure 2

Feynman Gate

This gate is also called as a controlled gate. Figure 2 shows a 2 x 2 gate i.e., it has 2 inputs and generates 2 outputs.It can be described as- inputs(A,B) and outputs P = A, Q = A XOR B. If we apply feynman gate double times, then we get back previous input. Quantum cost of Feynman Gate is 1. This gate is generally used for fan out purposes.

A	B	P	Q
0	0	0	0
0	1	0	1
1	0	1	1
1	1	1	0

Table 1

Truth Table of Feynman Gate

II. Toffoli Gate

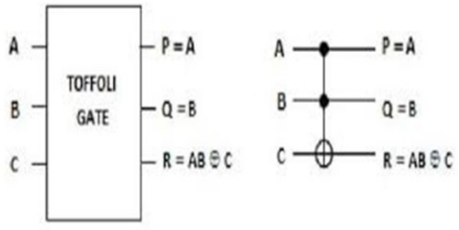


Figure 3

Toffoli Gate

Figure 3 shows a 3 x 3 gate fi.e., it has 3 inputs and generates 3 outputs.The input vector is I (A, B, C) and output is O (P, Q, R). It can be defined as- P = A, Q = B, R = AB XOR C. If we apply toffoli gate double times, then we get back previous input. Quantum cost of Feynman Gate is 5.

A	B	C	P	Q	R
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	0	0
1	0	1	1	0	1
1	1	0	1	1	1
1	1	1	1	1	0

Table 2

Truth Table of Toffoli Gate

III. Fredkin Gate

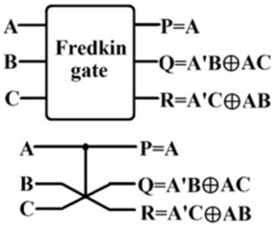


Figure 4

Fredkin Gate

Figure 4 shows a 3 x 3 gate i.e., it has 3 inputs and generates 3 outputs.The input vector is I (A, B, C) and output is O (P, Q, R). It can be defined as begin P = A, Q = A'B XOR AC, R = A'C XOR AB .This gate swaps the last two bits if and only if, the first bit is 1. Quantum cost of Fredkin Gate is 5.

A	B	C	P	Q	R
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	0	0
1	0	1	1	1	0
1	1	0	1	0	1
1	1	1	1	1	1

Table 3

Truth Table of Fredkin Gate

IV. Peres Gate

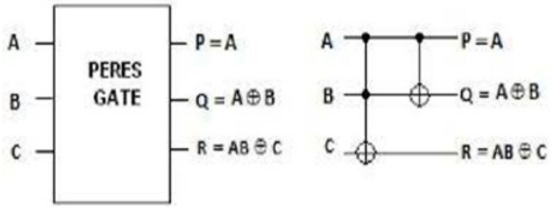


Figure 5

Peres Gate

Figure 5 shows a 3 x 3 gate i.e., it has 3 inputs and generates 3 outputs.The input vector is I (A, B, C) and output is O (P, Q, R). It can be defined as begin P = A, Q = A XOR B, R = AB XOR C . Quantum cost of Peres Gate is 4.

A	B	C	P	Q	R
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	1	0
1	0	1	1	1	1
1	1	0	1	0	1
1	1	1	1	0	0

Table 4

Truth Table of Peres Gate

V. BJN Gate

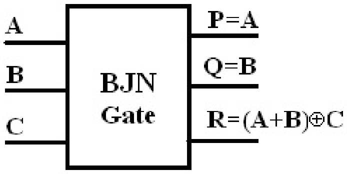


Figure 6

BJN Gate

Figure 6 shows a 3 x 3 gate i.e., it has 3 inputs and generates 3 outputs.The input vector is I (A, B, C) and output is O (P, Q, R). It can be defined as begin P = A, Q = B, R = (A+B) XOR C . Quantum cost of BJN Gate is 5.

A	B	C	P	Q	R
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	1
0	1	1	0	1	0
1	0	0	1	0	1
1	0	1	1	0	0
1	1	0	1	1	1
1	1	1	1	1	0

Table 5

Truth Table of BJN Gate

Reversible Gate Architecture of Toffoli Gate

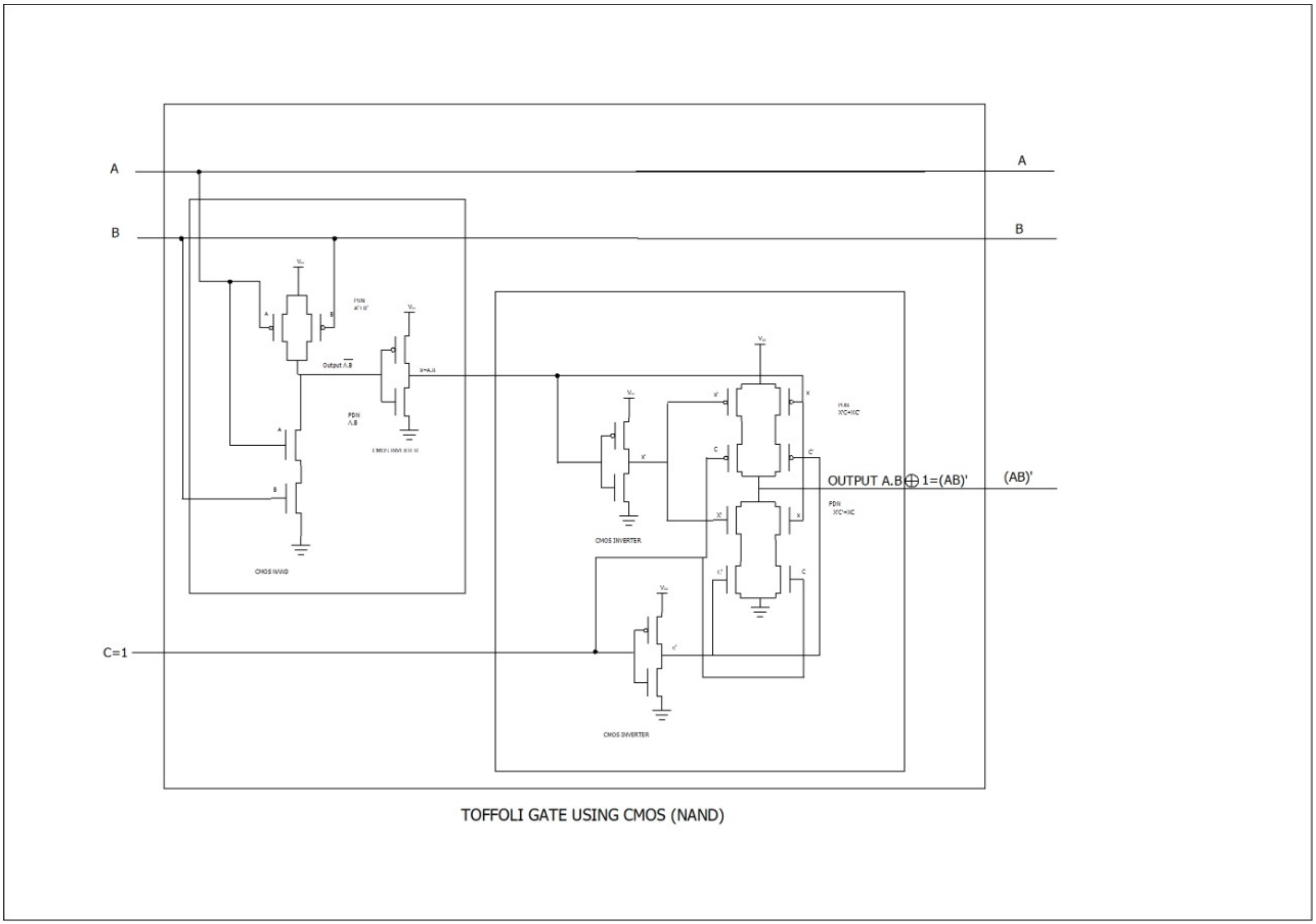


Figure 7

Diagram of Toffoli Gate using CMOS

Reversible Gate Architecture of Fredkin Gate

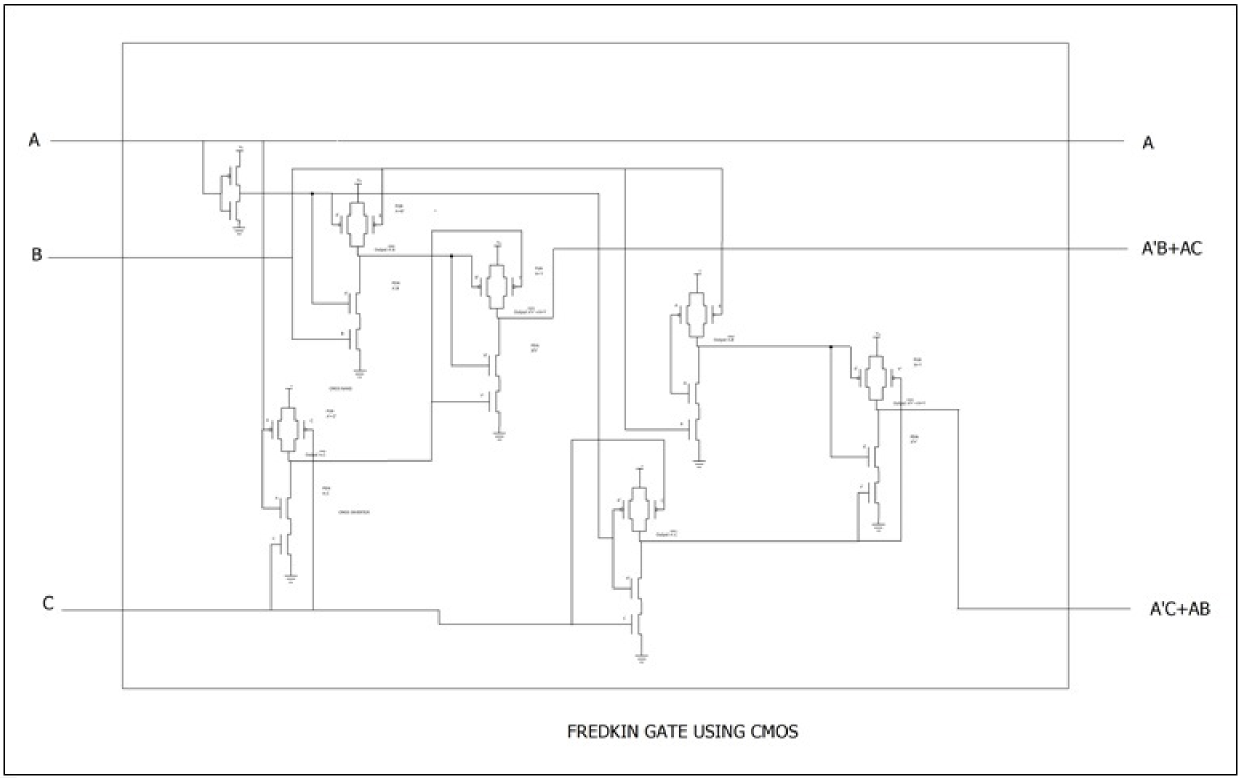


Figure 8

Diagram of Fredkin Gate using CMOS

Reversible Multiply Accumulate Unit

The proposed Reversible Multiply Accumulate Unit(MAC Unit) is based on the Booth's Multiplication Algorithm, This algorithm is a well-known algorithm to compute the multiplication of two numbers represented in 2's Complement number format. The procedure of the algorithm is enclosed in the flowchart in figure 14.

Booth's Multiplication

The proposed Reversible MAC unit consists of several components: Adder/Subtractor, 4 to 1 Multiplexer, D Flip-Flop, Universal/Bidirectional Information Shift Register implemented using Reversible gates. First we've discussed these components below.

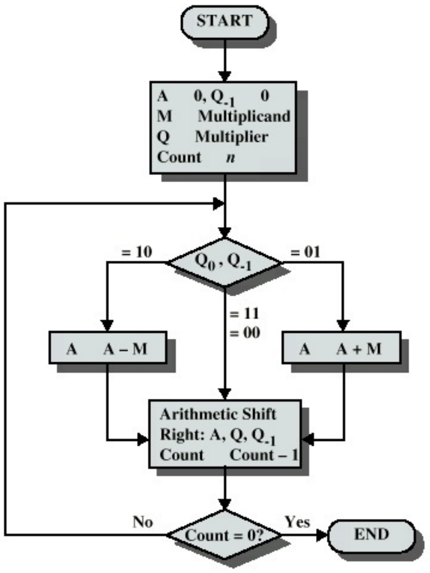


Figure 9

Flowchart of Booth's Algorithm

Adder/Subtractor using Reversible Gates

In the following figure 10, the diagram of proposed ADDER/SUBTRACTOR which is implemented using reversible Feynman and Toffoli gates is given. It takes three inputs Ai, Bi and Ci(Carry bit) and depending on the value of CTRL performs addition or subtraction. If CTRL=0 it works as a ADDER and for CTRL=1 it works as SUBTRACTOR. For initial C0, C0=CTRL is to be feed to work it as a ADDER/SUBTRACTOR with the help of the Feynman gate.The SUM or DIFFERENCE and Carry is generated with the help of Carry-Look-Ahead Adder logic. In first stage Pi=(Ai XOR Bi) is generated using Feynman gate second output and Gi'=(AiBi)' is generated using a Toffoli gate then Pi is feed with Ci in Feynman gate to generate Si/Di. To generate the Carry we need '(Ai XOR Bi)C', we generated the complement of this by feeding Pi,Ci with logic 1 in the Toffoli gate. Then the Carry is produced by feeding this with Gi' and logic 1 in Toffoli gate. Hence, our desired Carry=(Ai XOR Bi)C+AB i.e. (Gi+PiCi) is produced.

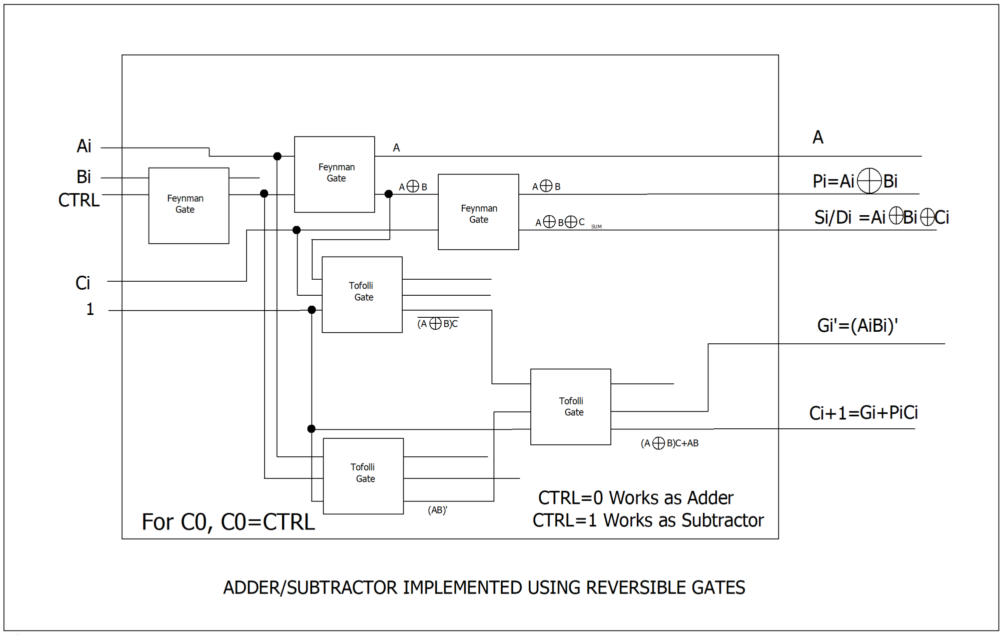


Figure 10

Proposed ADDER/SUBTRACTOR using Reversible Gates

The Quantum cost of this module is 18. The number of garbage outputs of this Adder/Subtractor is 7.

Reversible 4 to 1 MUX

The 4 to 1 reversible mux can be realized using [Gugnani and Kumar (2013)] Fredkin gate, Feynman gate, Peres gate and BJN gate. All the said gates have been explained above. The circuit diagram for 4 to 1 reversible gate is shown below in the figure 8. Here s1 and s0 are two select lines used. The outputs from g1 to g10 represent the garbage outputs. I0, I1, I2, I3 represents the inputs used. This module requires 24 quantum cost. And the number of garbage outputs of this 4X1 Reversible MUX is 10.

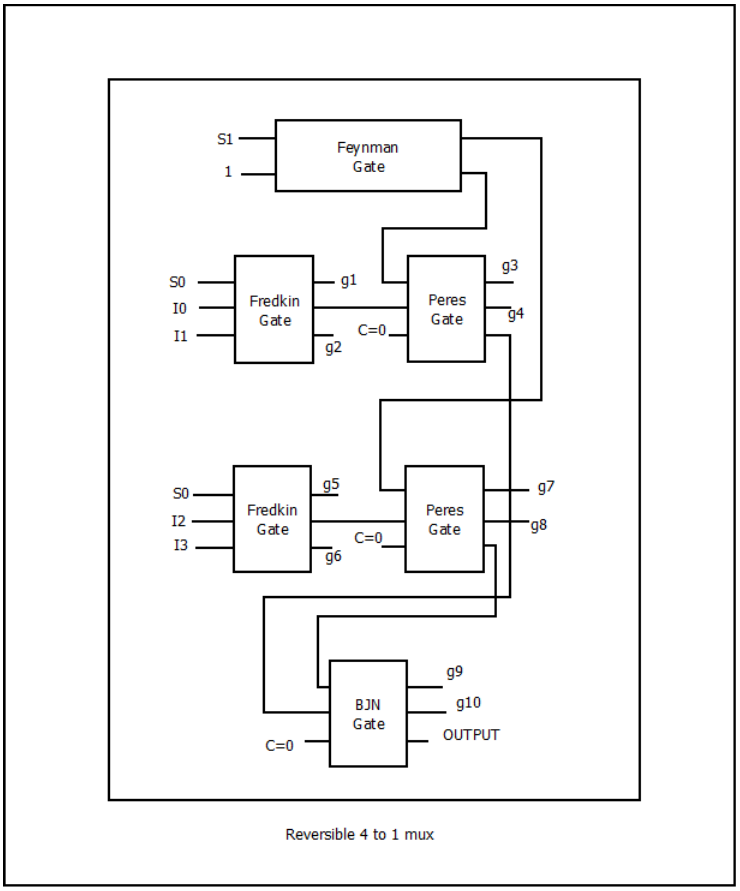


Figure 11

Reversible 4 to 1 Mux

Reversible D Flip-Flop

Reversible realizations of level-triggered D flip-flops [Guruprasad and Bhagat (n.d.)] are shown in Fig. 12. The state output is copied using a Feynman gate and fed back to the second input of the Fredkin gate. When the clock is one, then the feedback is connected to the state output maintaining the state output unchanged. When Clock becomes zero, then the D input is connected to the state output performing the level-triggered load operation. This realization requires 6 quantum costs and two garbage output.

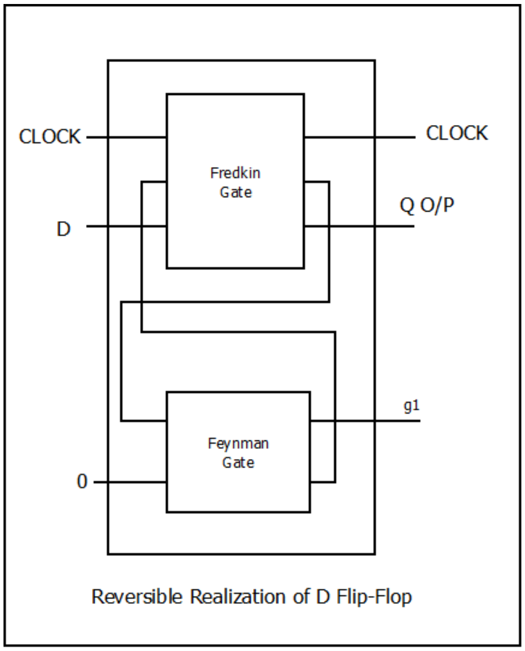


Figure 12

Reversible Realization of D Flip-Flop

Reversible Information Shifter

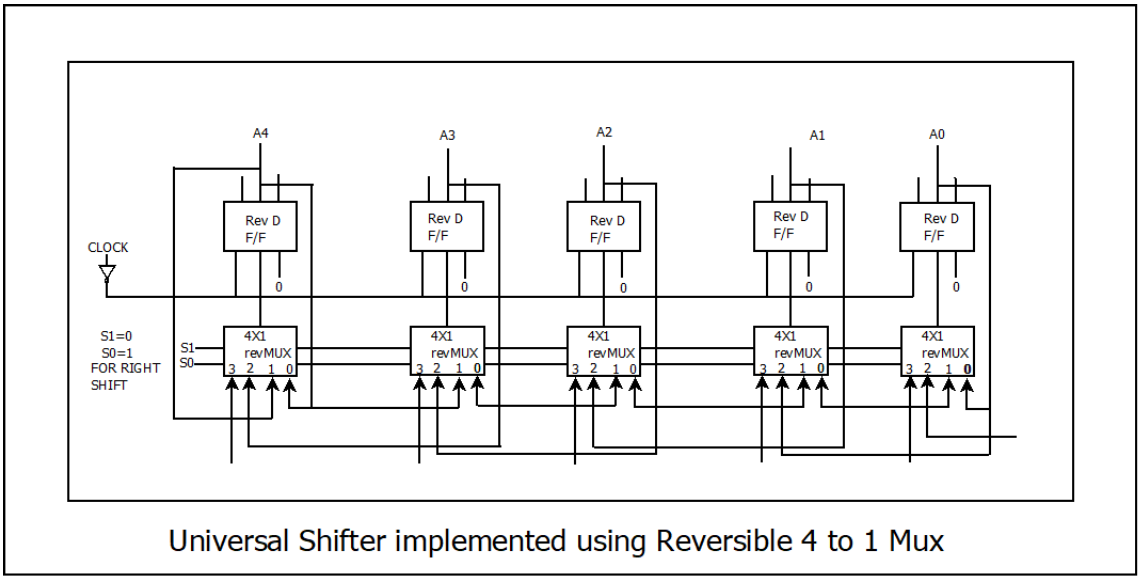


Figure 13

Universal Shifter implemented using Reversible MUX

In the above figure 13, there's a 5-bit universal information shifter implemented using reversible 4 to 1 mux and reversible D Flip-flop discussed earlier. This module can work as bidirectional shifter i.e. it can shift the data bits both from left to right and right to left. Depending on the select line values S1 and S0 it can work four functions: No change(00), Shift Right(01), Shift Left(10) and Parallel load(11). For our Booth's Reversible Multiply Accumulate Unit only the shift right operation is needed, hence the select lines we've used is always 01. As this is a 5-bit information shifter, we have to use 5 MUXes, and 5 D F/Fs. So, the total quantum cost for this 5-bit shifter is 150 with 60 number of garbage outputs. (MUX: 5*24; D: 5*6)

Reversible Booth's Multiplier

The proposed 2-bit Reversible Multiplier Accumulate Unit(MAC) in figure 14 consists of the several reversible modules discussed so far. It works on the Booth's Multiplication approach. So, it has named to Reversible Booth's MAC unit. The algorithm of the Booth's Multiplication is enclosed in the flowchart in figure 14(page). In the following figure 19, a 2-bit Booth's MAC unit is proposed, which takes two 2-bit inputs: Q1Q0(Multiplier) and M1M0(Multiplicand) and after successful completion of two clock cycles it produces a 4-bit Multiplication output at the right side with the output lines A1A0Q1Q0.

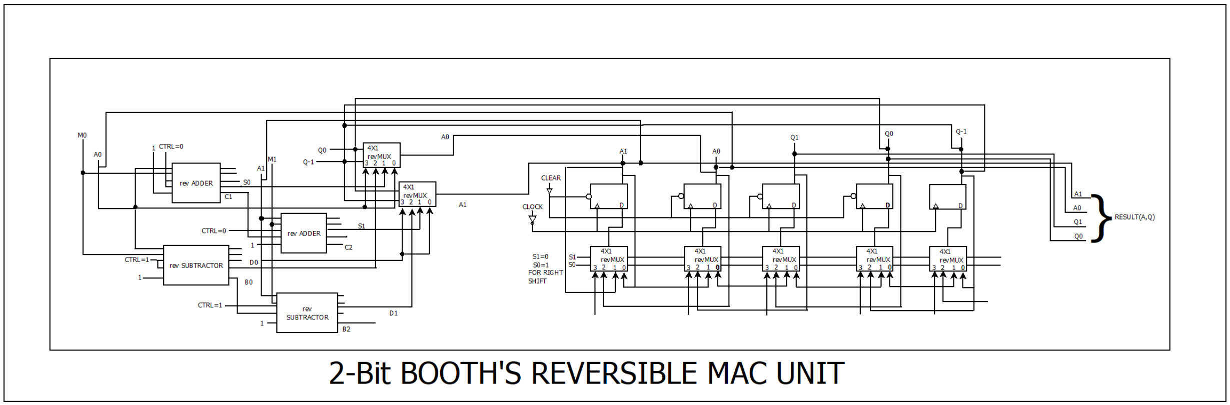


Figure 14

Proposed Booths Hardware Reversible Module

The multiplier and multiplicand needs to be in 2's Complement number representation format to produce the desired multiplication result. This module first generates the SUM and Difference of A and M using the reversible Adder/Subtractor. Based on the value of Q0Q-1 desired A1A0 is passed to the Shift Register using the Reversible 4 to 1 MUX. Then the reversible universal shift register right-shifts the values of A1A0Q1Q0Q-1 by one bit using S1S0(01) select lines. As it is using 2-bit information, according to Booth's Multiplication we need to perform this whole operation two times, so from the output of the shift register the values are passed on to the input lines so that we can perform the operation one more time. This Module has to be controlled by a hardware or software instruction to perform the Addition/Subtraction,Shifting two times sequentially.(We can use a timer/counter to perform it n-times for n-bit architecture) Before starting the multiplication, all the input lines need to be reset to 0. After 2 clock cycles, the output result of the multiplication is found at the output lines A1A0Q1Q0. The quantum cost of the 2-bit Reversible Booth's Multiply Accumulate Unit is 268 with 116 garbage output lines.

Result

The modules we have built/used so far have quantum cost and the number of garbage outputs of the modules we have used so far is as below:

Reversible Module	Quantum Cost	Garbage Outputs
Adder/Subtractor	18	7
D Flip-Flop	6	2
Multiplexer	24	10
Universal Shifter	150	60
Booth's MAC Unit	268	116

Table 6

Quantum cost and Garbage outputs of Reversible Modules

We have tried to simulate this same process in a finite machine(64-bit) with data loading and cleaning be in reversible manner(i.e. XORed) and run on JAVA environment. The process is enclosed in flowchart below.

The Output of the Java Program is attached below. Also the source code of the java program can be found here: `https://drive.google.com/file/d/1eAVRjszLy5IEQ7xV53QnE3PnXVK2J1BD/view?usp=sharing`

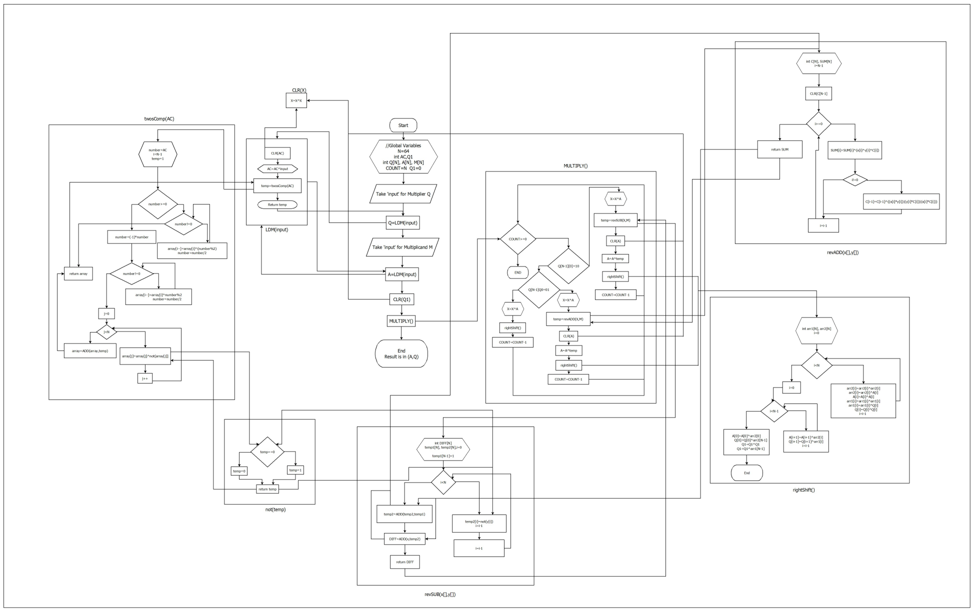
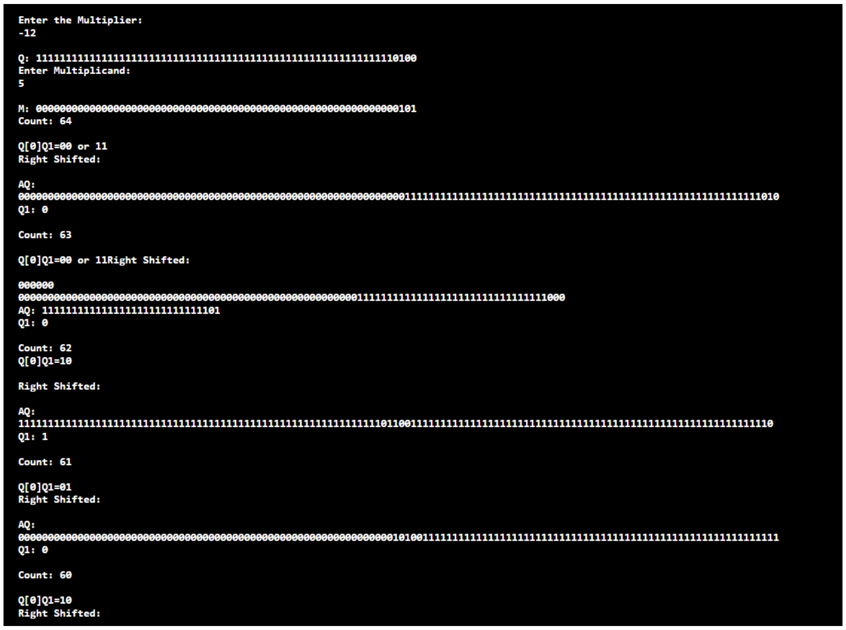


Figure 15

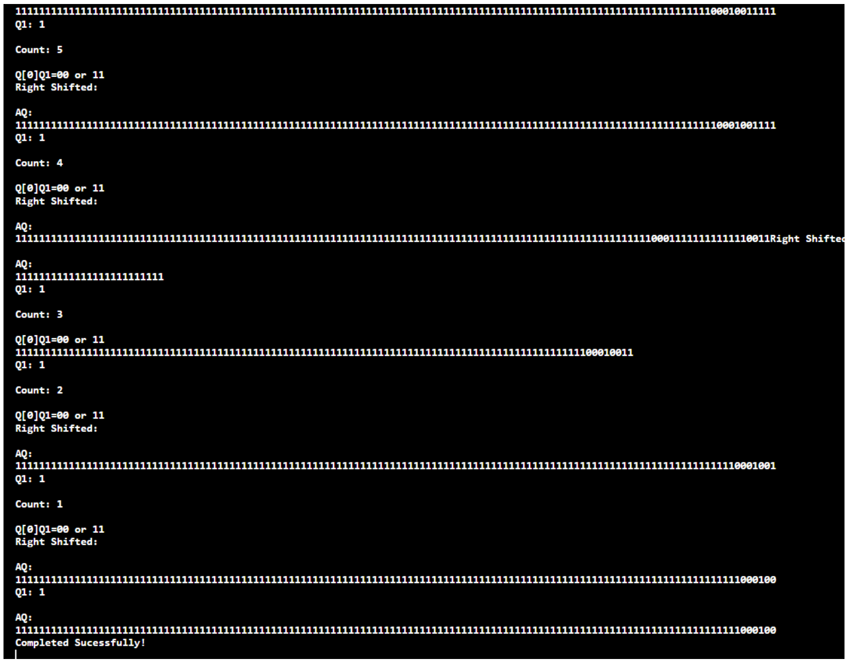
Flowchart of Booth's Multiplication with Reversible Data Loading and Cleaning



```
Enter the Multiplier:
-12

Q: 1111111111111111111111111111111111111111111111111111111111110100
Enter Multiplicand:
5

M: 0000000000000000000000000000000000000000000000000000000000000101
Count: 64

Q[0]Q1=00 or 11
Right Shifted:

AQ:
00000000000000000000000000000000000000000000000000000000000000000111111111111111111111111111111111111111111111111111111111111010
Q1: 0

Count: 63

Q[0]Q1=00 or 11Right Shifted:

000000
00000000000000000000000000000000000000000000000000000000011111111111111111111111111111111000
AQ: 111111111111111111111111111101
Q1: 0

Count: 62
Q[0]Q1=10

Right Shifted:

AQ:
1111111111111111111111111111111111111111111111111111111111111011001111111111111111111111111111111111111111111111111111111111110
Q1: 1

Count: 61

Q[0]Q1=01
Right Shifted:

AQ:
00000000000000000000000000000000000000000000000000000000000000010100111111111111111111111111111111111111111111111111111111111111
Q1: 0

Count: 60

Q[0]Q1=10
Right Shifted:
```

Figure 16

Output of Flowchart implementation in a Java Program



```
11111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111100010011111
Q1: 1

Count: 5

Q[0]Q1=00 or 11
Right Shifted:

AQ:
11111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111110001001111
Q1: 1

Count: 4

Q[0]Q1=00 or 11
Right Shifted:

AQ:
1111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111100011111111111110011Right Shifte

AQ:
1111111111111111111111111
Q1: 1

Count: 3

Q[0]Q1=00 or 11
11111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111100010011
Q1: 1

Count: 2

Q[0]Q1=00 or 11
Right Shifted:

AQ:
11111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111110001001
Q1: 1

Count: 1

Q[0]Q1=00 or 11
Right Shifted:

AQ:
11111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111000100
Q1: 1

AQ:
11111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111000100
Completed Sucessfully!
```

Figure 17

Output of Flowchart implementation in a Java Program

Input from user: 12 and -5

12 is represented as 0000000000000000000000000000000000000000000000000000000000001100

in 2's Complement

-5 is represented as 1111111111111111111111111111111111111111111111111111111111111011

Bit Architecture chosen here is 64-bit.

Output should be: -60 which is

11111111111111111111111111111111111111111111111111111111111111111111

111111111111111111111111111111111111111111111111111111000100

in 64 bit 2's complement representation.

Conclusion

We have developed the Reversible Multiply Accumulate Unit using 2's complement Booth's Multiplication technique. While developing the MAC unit, we need to use Adder, Subtractor, Multiplexer and Right Shifter. For this, we have built a Adder/Subtractor using reversible gates(figure 10), used a 4X1 Reversible Multiplexer(figure 11), and Universal Information Shifter(figure 13) which uses the MUX(figure 11) and Reversible D Flip-Flop(figure 12). The Booth's Reversible MAC unit(figure 14) is based on 2-bit Architecture, it can be easily built for desired bit architecture. We can minimize the quantum cost of the module if we can replace the consisting modules like ADDER/SUBTRACTOR, Shifter with less quantam cost modules. Also, this MAC module is dependent on external timer/counter/clock to run it 2-times and also externally dependent for cleaning/resetting of it's input lines for further usage. This module is an attachment which can be used in DSP or ALU for multiplicative arithmetic operations. This module is capable of doing integer multiplication only if represented in 2's Compliment Number Format and has a drawback for not being able to solve floating point arithmetic multiplication. Also we didn't enclosed the power consumption of the modules, as we were unable to fabricate it in lack of semiconductor unavailability.

References

Das, S. K., Kanhe, A., & Talwekar, R. (2013). Design and implementation of high-performance mac unit. *International Journal of Scientific & Engineering Research*, *4*(6).

Frank, M. P. (2005). Introduction to reversible computing: motivation, progress, and challenges. In *Proceedings of the 2nd conference on computing frontiers* (pp. 385–390).

Gugnani, S., & Kumar, A. (2013). Synthesis of reversible multiplexers. *International Journal of Scientific & Engineering Research*, *4*(7), 1859–1863.

Guruprasad, K., & Bhagat, S. K. (n.d.). An algorithm to reduce quantum cost and garbage outputs in reversible logic circuits.

Malik, S., & Dhall, S. (2012). Implementation of mac unit using boothmultiplier & ripple carry adder. *International Journal of Applied Engineering Research*, *7*(11), 358–363.