2022 IEEE 19th India Council International Conference (INDICON) | 978-1-6654-7350-7/22/$31.00 ©2022 IEEE | DOI: 10.1109/INDICON56171.2022.10040039

FPGA Based Resource Efficient Simulation and Emulation Of Grover's Search Algorithm

1st Kingsuk Bag
Department of ECE
IIIT Allahabad
Prayagraj, India
kbag7890@gmail.com

2nd Manish Goswami
Department of ECE
IIIT Allahabad
Prayagraj, India
manishgoswami@iiita.ac.in

3rd Kavindra Kandpal
Department of ECE
IIIT Allahabad
Prayagraj, India
kavindra@iiita.ac.in

***Abstract* - The introduction of quantum computing has led to an immense increase in performance of various search and encryption algorithms. However the practical realization of quantum computers is[1] a challenging task. Hence quantum algorithms are being modeled on classical platforms. Hardware emulation of quantum computers has been found to be more effective than software simulations. In this paper, Grover's search algorithm has been modeled in the classical platform using both hardware emulation and software simulation approaches. The modified architecture of the field programmable gate arrays (FPGAs) and the unique data path implemented in this paper results in considerable speed-up and lesser utilization of resources as compared to other existing works. This work includes both simulation and emulation of Grover's search for 2,3 and 5 qubit systems. The novel emulation approach has led to a 10^4 times reduction in time of search than the corresponding simulation.**

Keywords: Grover's search algorithm, qubits, quantum computing, FPGA, architecture, simulation, emulation

I. INTRODUCTION

The concept of quantum computing has been around for quite some time now. The basic idea of computations adhering to quantum principles was proposed way back in 1982 by Richard Feynman[1][2]. Since then Quantum computing has come a long way and many advances have been made in this domain. Development of new and advanced quantum computing algorithms have bolstered the amount of ongoing research in this domain. These quantum algorithms utilize the properties of quantum mechanics such as entanglement and superposition which results in a rampant speedup as compared to its classical counterparts. Quantum computers have made promises of solving many practical problems which might not be possible by using the existing classical computers. However, the real physical realization of quantum computers which could be commercialized has been extremely difficult[3][4]. Research is still in progress in this regard. Keeping in mind this current scenario, there have been many attempts to mimic the workings of a quantum computer using the existent technologies of classical computing. However the modern day classical computers are based on the Von Neumann architecture which is essentially serial in nature and trying to emulate the quantum computers on these machines defeat the purpose of achieving the parallelism for which the quantum computers are regarded. Therefore, a field programmable gate array (FPGA) could be used to mimic the parallel processing attributes of the quantum computer as the hardware in the FPGA could be emulated according to our needs.

II. LITERATURE REVIEW

Since the inception of quantum computing, researchers have always looked into alternate implementations of the quantum computer due to the difficulties in the realization of a physical quantum computer. However modeling the quantum computing framework on a classical platform has always been a challenge for researchers due to the non-intuitive nature of the quantum computers and the shortage of resources that are present in the classical computing platforms while trying to implement higher qubit systems. Similar sort of challenges also occur in case of FPGA as this too is constrained by the total number of Flip flops and LUTs that are available on the particular FPGA. A careful and thrifty use of these resources and precise manipulation of the quantum computing algorithms is needed in order to successfully implement them on the FPGA.

In the decades gone by ,many researchers have put forward various theories on the implementation of different quantum computing algorithms on a FPGA. However most of these papers[5-7] lack in providing the critical design attributes which actually act as the

[1] **978-1-6654-7350-7/22/$31.00 ©2022 IEEE**

bridge between the classical FPGA and an original quantum system. Moreover there is also a lack of comparison between various implemented forms of the algorithm in regards to the efficiency and the amount of resources used. Some implementations of algorithms that have been mapped in the past are shown to support a lower number of qubit systems only. They typically fail when the number of qubits are increased as the implementation fails to manage the need for exponential growth in required resources. The majority of the proposed works till now have been derived from the quantum circuit model. This model describes the basic quantum gates and uses these to form a digital-like circuit which implements the required quantum algorithms. However the disadvantage with this approach is that it leads to more utilization of resources and lesser efficiency. This approach was used in [6] to implement Grover's search algorithm.

A similar situation was also observed in [10] where the quantum circuit was constructed in such a way that it bound the input states to the system to be the computational basis only and the system failed when a superposition of the computational basis states was fed to the system. Some research works related to emulating the quantum computing framework on an FPGA have utilized a pipeline inspired architecture of the FPGA . According to the author et al this has led to a substantial increase of the throughput of the implemented algorithm along with decreasing the delay that is present in the critical path. But this approach is not fully suitable for this purpose as it fails to take into consideration the exponential need for resources with even a small increase in the number of qubits. Similar to the attempts of emulating the quantum computer, there have been multiple attempts to simulate the various quantum computing algorithms using various programming languages like C and python. Some of the stand out implementations include the development of an open source library called “libquantum” which has been presented in [8] and this supports some useful algorithms both as a special case and also in their generic state.

This research work focuses on the above mentioned shortcomings and makes certain changes in the implementation of the algorithm to consider the effects of entanglement and superposition. The attempted algorithm in this paper includes Grover’s search algorithm . Here we will target both simulation and emulation of the aforementioned algorithm. It will also compare the speedups of the emulation approach over the corresponding simulation approach thus demonstrating the speedup caused due to the intelligent manipulation of FPGA architecture. The resource utilization of FPGA will also be demonstrated along with the effects when the number of qubits is increased.

III. FUNDAMENTAL CONCEPTS

The characteristic of quantum computing which sets it apart from classical computation is that it is probabilistic in nature and deterministic. That is it has to deal with a number of additional issues like error in measurement and interpretation. To compensate for the errors related to such a probabilistic system when implemented on classical hardware, the approach of taking multiple observations has to be adopted to get better and more accurate results of the implemented algorithms. This is one of the factors which get kind of left out in contemporary research works. But the work presented here takes it into consideration.

In the world of classical computation, the smallest morsel of information is the classical bit. A bit can be present in either a state of being 0 or in the state of being 1 but not both simultaneously. This could be represented as matrices in the following way:

$$state - > 0 = \begin{vmatrix} 1 \\ 0 \end{vmatrix} \ldots\ldots\ldots\ldots(1)$$

$$state - > 1 = \begin{vmatrix} 0 \\ 1 \end{vmatrix} \ldots\ldots\ldots\ldots(2)$$

The counterpart of the classical bit is the quantum bit, which represents the smallest piece of information in the domain of quantum computation. The quantum bit is commonly referred to as the qubit. The qubit is a combination of the basic boolean states of 0 and 1 which implies that it could be both 0 and 1 at the same time. In a more general sense, it can be expressed as a superposition of the states of 1 and 0. The states if 1 and 0 are represented in the quantum domain by using the Dirac ket notation(“| 〉”). 0 and 1 are represented by their respective ket notations that is: $|0\rangle$ and $|1\rangle$.

A qubit represented as a superposition of $|0\rangle$ and $|1\rangle$ could be given by:

$$|\boldsymbol{\Psi}\rangle = \boldsymbol{\alpha}\,|0\rangle + \boldsymbol{\beta}\,|1\rangle = \begin{vmatrix} \alpha \\ \beta \end{vmatrix} \ldots\ldots\ldots\ldots(3)$$

In (3) both the constants α and β are complex numbers and the squares of their magnitudes represent the probability of the occurrence of their respective state once the measurement of the state is done. $|\alpha|^2$ represents the the probability that the bit is state $|0\rangle$ after measurement and similarly $|\beta|^2$

represents the the probability that the bit is state |1⟩ after measurement. Thus it can be concluded that $|\alpha|^2 + |\beta|^2 = 1$ as after measurement the bit becomes a classical bit which could take either 0 or 1. So there are only two possibilities and the sum of the two probabilities should be 1.The above representation was for a 2 qubit system and a similar approach is followed for representing a n qubit quantum state. For that purpose a total of 2^n constants will be required and the sum of the squares of their amplitudes should also be 1.

Classical computers operate via logic gates and wires. The logic gates perform operations on input bits whereas the wires carry the output bits from one gate to the other. Quantum gates are analogous to the classical logic gates in the quantum world. They have a basic circuit which operates on the input qubits to alter its state. However a major difference between these is that while most classical gates are non -reversible in nature, quantum gates are reversible, which means that we can apply the gate twice to get back the original state of the system where the operation initially began.

It has been observed that quantum operations could be represented by unitary matrices operating on the input qubits. The unitary matrices are reversible in nature that is multiplying them with their transpose leads to the identity matrix. So we always have a way of gaining back the original input and hence it satisfies the criteria that the gates should be reversible in nature. In general a n * n matrix is said to be unitary in nature if: $A\ A^{*T} = I_N$ i.e the product of A with its conjugate transpose leads to the identity matrix.

IV. IMPLEMENTATION

Grover's Search Algorithm:

One of the most commonly used algorithms in the field of computer science is a search algorithm. There are a wide variety of algorithms that are designed to search for a particular element in an array. However most of them work efficiently only when the array is sorted in either ascending or descending order. If it is not so, then the algorithm either does not work or it takes a lot of time to search for a particular element.

To remedy this problem, an algorithm was devised by Lov Grover in 1996 named the Grover's search algorithm. This is not a classical algorithm and is quantum in nature. It makes use of the quantum principles of superposition and entanglement to easily search for an element in an unsorted array.The unique feature of Grover's search algorithm is that it determines the presence of the element in the array in square root number of iterations over the classical approach. Expressing the same in big O notation, the complexity of Grover's search algorithm is given by $O(\sqrt{N})$ whereas the best classical algorithm takes at least O(N) to solve the same problem . Here N represents the total number of elements in the unsorted array.Another reason why Grover's search algorithm is often considered as a cornerstone of the quantum computing framework is because it can be modified slightly to form the amplitude amplification algorithm which opens up broad areas for its application. It provides an effective way for solving NP complete problems as these NP problems contain a sub process of searching for an element. It is also particularly useful for problems in quantum theory involving black box problems which are used to identify whether a given element is distinct in nature. The scope of the Grover's search algorithm can also be found in the field of cryptography. It provides a considerable amount of speedup over the existing algorithms in finding the inverse of a function, a feature which is used in cryptic key inversion techniques.

Mathematically speaking, grover's search algorithm is expressed as:

$\boldsymbol{f : \{0, 1\}^n \rightarrow \{0, 1\}}$ is sure to be found provided that there is a presence of such a binary string y_0 which satisfies the following:

$$\boldsymbol{f(y)=1 \text{ if } y = y_0 \; ; \quad \text{else,} \quad f(y)=0} \quad \ldots\ldots\ldots..(4)$$

The process flow of the Grover's search algorithm could be described in terms of the quantum circuit model.

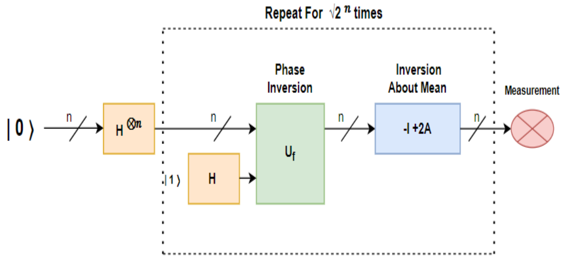


Fig 1- Quantum circuit model of Grover's search algorithm for n qubits

The basic operations involved include the inversion of the phase of the input qubits and the mean inversion of the qubits. In the above U_f represents the inversion of the phase while -I+2A represents the inversion about the mean.

The input qubits are first subjected to the N bit Hadamard operation which splits into basis states of equal probability. An extra qubit is set to 1 is used too in the process which is also subjected to the Hadamard operation before all the qubits are subjected to U_f for phase inversion. This is followed by the inversion about the mean. If the Grover's search is being performed on "n" elements, then the above described process is to be repeated for $\sqrt{2^n}$ times. Post this process, the measurement of the amplitudes of the qubits is to be done and the one element which is being searched for should have the highest amplitude thus leading to its detection.

A step by step process flow of the Grover's search algorithm is as given:

1. We start the process with $|\mathbf{0}\rangle$ as the input qubits which act as the target.
2. This target qubits are subjected to the n bit Hadamard operation, $H^{\otimes n}$

A loop is started which runs for $\sqrt{2^n}$ times:

a. The operation of phase inversion ,$U_f\,(I \otimes H)$ is applied to the input.

b. The operation of inversion about mean . -I+2A is applied to the input from the previous operation.

3. Loop ends and qubits are measured

The quantum circuit model approach of the Grover's search algorithm is used in the simulation of the algorithm. However it involves operation with large matrices and hence it leads to problems when this approach is followed in FPGA as the FPGA is resource constrained. So a different approach is used for emulation of the algorithm. While modeling the Grover's search algorithm in the FPGA we will not directly resort to the application of the Quantum circuit model in order to conserve resources of the FPGA. Here a mathematical model consisting of arithmetic and logical operations is used to mimic the quantum circuit model but with considerably less memory and storage. This leads to a significant speed up as well as more resource utilization and lower power consumption. A comparative study of both the approaches have been provided in later sections.This approach utilizes different smaller modules which are first created. Modules such as comparator, adder, subtracter , multiplier , multiplexer and demultiplexer are individually constructed and are called into the main algorithm ensuring reusability of the design. The basic idea is still based on the two main operations of the algorithm, namely, the inversion of phase and the inversion about the mean.The data path model of the FPGA implementation has been described via the illustration given below:

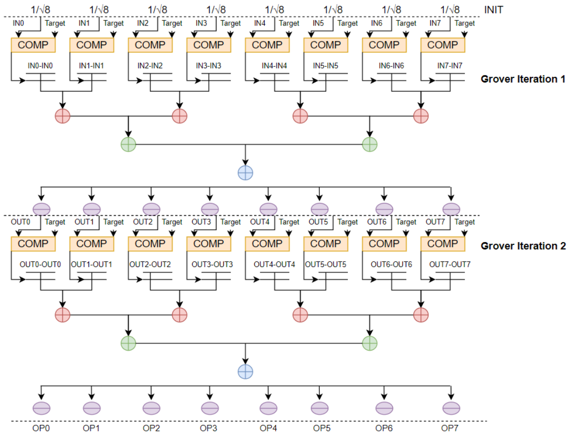
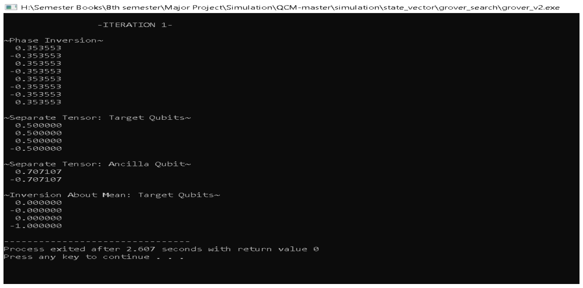


Fig 2- Data path model of FPGA implementation of Grover's search algorithm for 3 qubits

The above diagram describes the data path for a 3 bit Grover's search. The other searches for various numbers of qubits are based on the same principle. As evident from the Fig, the input quantum state is set to values of equal probability initially. This is done through the INIT function which performs the said operation. This is followed by the COMP module which compares the values of the input qubits with the target qubits. Then the desired phase inversion and the inversion about the mean are performed using simple and arithmetic and logical operations instead of using complex matrices which require a large number of registers to store.

V. SIMULATION RESULTS

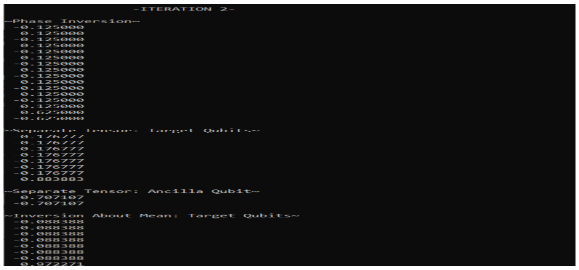
The simulation of the 2,3 and 5 qubit systems have been performed using C programming language and compiled using Dev C software. The results of the simulation are illustrated in this section.



H:\Semester Books\8th semester\Major Project\Simulation\QCM-master\simulation\state_vector\grover_search\grover_v2.exe

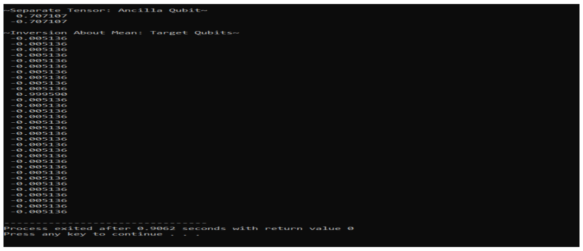
```
                 -ITERATION 1-

~Phase Inversion~
  0.353553
 -0.353553
  0.353553
 -0.353553
  0.353553
 -0.353553
 -0.353553
  0.353553

~Separate Tensor: Target Qubits~
  0.500000
  0.500000
  0.500000
 -0.500000

~Separate Tensor: Ancilla Qubit~
  0.707107
 -0.707107

~Inversion About Mean: Target Qubits~
  0.000000
 -0.000000
  0.000000
 -1.000000

--------------------------------
Process exited after 2.607 seconds with return value 0
Press any key to continue . . .
```

Fig 3- Software simulation of 2 qubit Grover's search algorithm for searching 3



```
                    -ITERATION 2-
~Phase Inversion~
 -0.125000
  0.125000
 -0.125000
  0.125000
 -0.125000
  0.125000
 -0.125000
  0.125000
 -0.125000
  0.125000
 -0.125000
  0.125000
 -0.125000
  0.125000
  0.625000
 -0.625000

~Separate Tensor: Target Qubits~
 -0.176777
 -0.176777
 -0.176777
 -0.176777
 -0.176777
 -0.176777
 -0.176777
  0.883883

~Separate Tensor: Ancilla Qubit~
  0.707107
 -0.707107

~Inversion About Mean: Target Qubits~
 -0.088388
 -0.088388
 -0.088388
 -0.088388
 -0.088388
 -0.088388
 -0.088388
  0.972271
```

Fig 4- Software Simulation Of 3 qubit Grover's search algorithm for searching 7



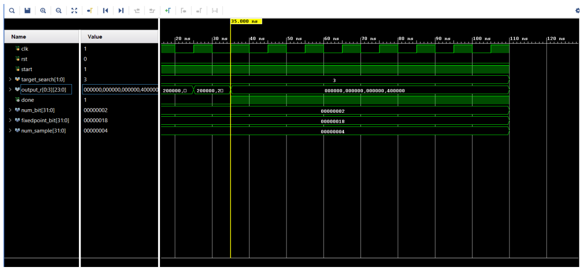
```
~Separate Tensor: Ancilla Qubit~
  0.707107
 -0.707107

~Inversion About Mean: Target Qubits~
 -0.005136
 -0.005136
 -0.005136
 -0.005136
 -0.005136
 -0.005136
 -0.005136
 -0.005136
 -0.005136
 -0.005136
  0.999590
 -0.005136
 -0.005136
 -0.005136
 -0.005136
 -0.005136
 -0.005136
 -0.005136
 -0.005136
 -0.005136
 -0.005136
 -0.005136
 -0.005136
 -0.005136
 -0.005136
 -0.005136
 -0.005136
 -0.005136
 -0.005136
 -0.005136
 -0.005136
 -0.005136

--------------------------------
Process exited after 0.9062 seconds with return value 0
Press any key to continue . . .
```

Fig 5- Software simulation of 5 qubit Grover's search algorithm for searching 10

It was observed that the simulation time taken was in the order of milliseconds.

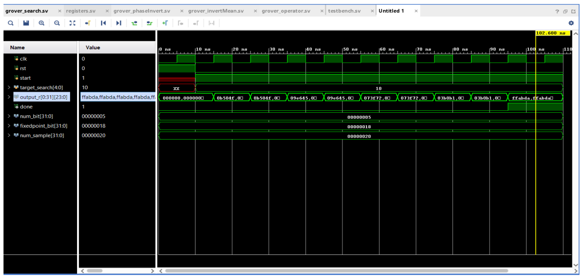
VI. EMULATION RESULTS

The emulation of the 2,3 and 5 qubit systems have been performed using System verilog and compiled using Xilinx Vivado HLS. The results of the emulation are represented in this section.



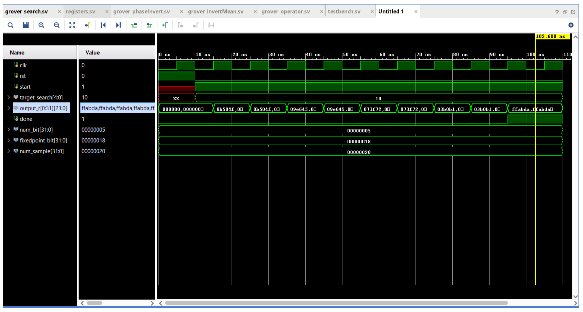
Name	Value
clk	1
rst	0
start	1
target_search[1:0]	3
output_r[0:3][23:0]	000000,000000,000000,400000
done	1
num_bit[31:0]	00000002
fixedpoint_bit[31:0]	00000018
num_sample[31:0]	00000004

Fig 6- Emulation results Of 2 qubit Grover's search algorithm for searching 3



Name	Value
clk	0
rst	0
start	1
target_search[4:0]	10
output_r[0:31][23:0]	ffabda,ffabda,ffabda,ffabda,ff
done	1
num_bit[31:0]	00000005
fixedpoint_bit[31:0]	00000018
num_sample[31:0]	00000020

Fig 7- Emulation results Of 3 qubit Grover's search algorithm for searching 7



Name	Value
clk	0
rst	0
start	1
target_search[4:0]	10
output_r[0:31][23:0]	ffabda,ffabda,ffabda,ffabda,ff
done	1
num_bit[31:0]	00000005
fixedpoint_bit[31:0]	00000018
num_sample[31:0]	00000020

Fig 8- Emulation results Of 5 qubit Grover's search algorithm for searching 10

It was observed that the emulation time taken was in the order of nanoseconds with a 150 MHz clock frequency.

VII. COMPARISON RESULTS

The section mentioned below provides graphical representations of various parameters and also compares the work presented in this report with existing works.[3][5][8].

Fig 9 provides us a bird's eye view of the number of logic registers of the Artrix 7 FPGA that have been used to perform the search for various numbers of qubits. As expected the number of registers increased with increased precision for all of the 2,3 and 5 qubit systems.

The work presented in [3] uses a concurrent architecture , [5] is based on pipeline architecture and [8] uses a serial-parallel architecture. As illustrated by Fig 10, the Hybrid architecture used in this research clearly outperforms all of the above mentioned ones by reducing the resource utilization in a significant manner.

A tradeoff exists between the maximum operating frequency achievable and the number of resources that are being utilized to perform the search.While comparing the maximum operating frequency in Fig 11 the hybrid architecture outperforms all other architectures except the pipelined version[5] , but this is more than compensated by the lesser utilization of resources by the hybrid architecture.

Fig 12 gives an idea about the amount of speedup that was achieved in the emulation approach as compared to the simulation approach. It is observed that the speedup was maximum for the 2 qubit system and it reduced as the number of qubits increased. This is due to the fact that the FPGA is after all resource constrained . However in all the three cases a speed up of the order of 10^4 was achieved.

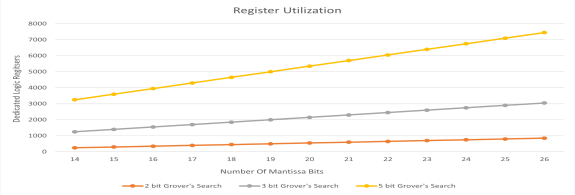


Fig 9: FPGA resource utilization for varying Mantissa bits

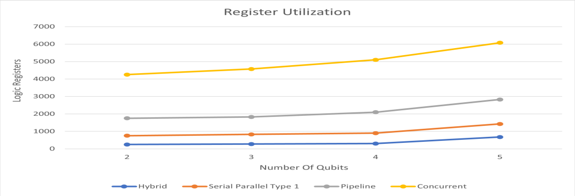


Fig 10:FPGA resource utilization for various architectures

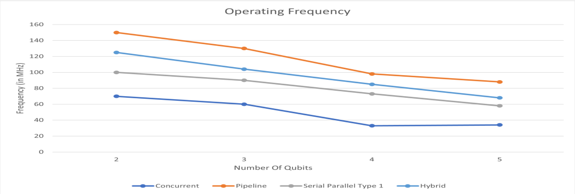


Fig 11:Operating frequency for various architectures[3][5][8]

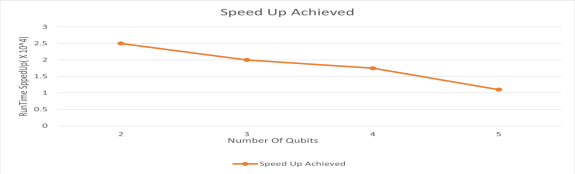


Fig 12: Comparison of speed-up of emulation Vs simulation for changing qubits

VIII. CONCLUSION

This research work has focused on making certain changes in the implementation of the Grover's search algorithm to consider the effects of entanglement and superposition. It has taken into account both the simulation and emulation of the aforementioned algorithm. Three different kinds of systems have been considered namely the 2,3 and 5 qubit systems and searches were performed for arbitrary numbers on these. The simulation results of the iterations have been attached in the report. The emulation results for the same consist of the output waveforms, resource utilization of FPGA, power consumption analysis and timing analysis of individual systems.The architecture followed in this paper has resulted in a significant improvement in the resource utilization as compared to other existing research works in this domain. Implementation of the hybrid architecture along with the novel data path model has led to a ten fold decrease in resources utilized as well as 10^4 times decrease in the time taken for a search as compared to the simulation approach.

IX. REFERENCES:

[1] N. Mahmud and E. El-Araby, "A Scalable High-Precision and High-Throughput Architecture for Emulation of Quantum Algorithms," 2018 31st IEEE International System-on-Chip Conference (SOCC), 2018, pp. 206-212, doi: 10.1109/SOCC.2018.8618545.

[2] N. Benchasattabuse, P. Chongstitvatana and C. Apomtewan, "Quantum Rough Counting and Its Application to Grover's Search Algorithm," 2018 3rd International Conference on Computer and Communication Systems (ICCCS), 2018, pp. 21-24, doi: 10.1109/CCOMS.2018.8463331

[3] H. Li and Y. Pang, "FPGA-Accelerated Quantum Computing Emulation and Quantum Key Distillation," in IEEE Micro,2021, doi: 10.1109/MM.2021.3085431.

[4] N. Mahmud and E. El-Araby, "Towards Higher Scalability of Quantum Hardware Emulation Using Efficient Resource Scheduling," 2018 IEEE International Conference on Rebooting Computing (ICRC), 2018, pp. 1-10, doi: 10.1109/ICRC.2018.8638610.

[5] V. Hlukhov, "FPGA Based Digital Quantum Computer Verification," 2020 IEEE 11th International Conference on Dependable Systems, Services and Technologies (DESSERT), 2020, pp. 178-182

[6] S. Yang, Z. Lu and Y. Li, "High-Speed Post-Processing in Continuous-Variable Quantum Key Distribution Based on FPGA Implementation," in Journal of Lightwave Technology, 2020, doi: 10.1109/JLT.2020.2985408.

[7] M. Aminian, M. Saeedi, M. S. Zamani and M. Sedighi, "FPGA-Based Circuit Model Emulation of Quantum Algorithms," 2018 IEEE Computer Society Annual Symposium on VLSI, 2008, pp. 399-404, doi: 10.1109/ISVLSI.2008.43.

[8] J. Chen, L. Wang and B. Wang, "Quantum FPGA architecture design," 2013 International Conference on Field-Programmable Technology (FPT), 2013, pp. 354-357, doi: 10.1109/FPT.2013.6718386.

[9] A. G. Mazăre, L. Mihai Ionescu, I. Liță, G. Șerban and N. Belu, "New FPGA design solution using quantum computation concepts," 2021 IEEE 27th International Symposium for Design and Technology in Electronic Packaging (SIITME), 2021, pp.388-391,doi:10.1109/SIITME53254.2021.966365.

[10] K. Kang, "Two improvements in Grover's algorithm," The 27th Chinese Control and Decision Conference (2015 CCDC), 2015, pp. 1179-1182, doi: 10.1109/CCDC.2015.7162096.