Reducing Computational Complexity of Mathematical Functions Using FPGA

Neha Gour[1]*, Arup Banerjee[2]

[1]*Banasthali University, Banasthali-304022*

[2]*HRDS, Raja Ramanna Centre for Advanced Technology, Indore-452013*

*E-mail: nehagaur339@gmail.com

ABSTRACT

- Analysis of complex algorithm demands for less execution time and low storage space.
- This study aims to reduce the time complexity (execution time) using parallel computing of developed algorithms.
- These algorithms are developed for executing computationally intensive applications such as weather forecasting.
- Objective of this work is to improve the execution time of mathematical functions using the concept of concurrency.
- Parallelism is explored by the optimization directives such as loop unrolling and loop pipelining to improve the performance of system.
- Experiments for single loop addition were done using loop pipelining and loop unrolling which shows the reduction in time delay of approx. 28% and 71% respectively as compared to conventional processing.

INTRODUCTION

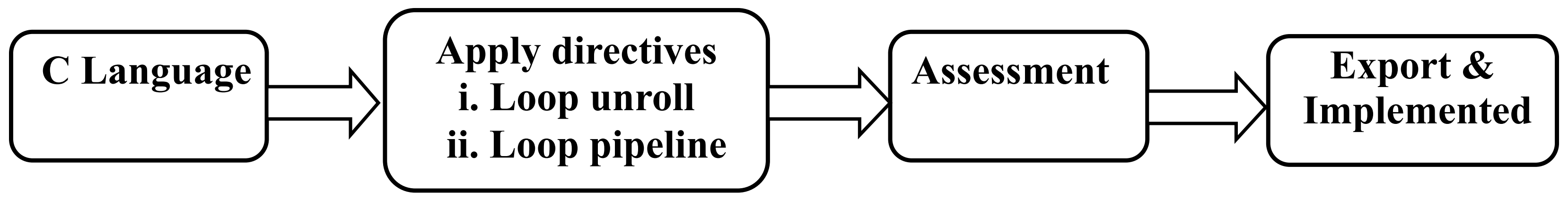
- Conventional processors exhibit sequential processing thus they need lot of time to execute instructions.
- Parallel processing through FPGA is one possible solution to reduce execution time.
- Parallel processing through FPGA is implemented using Vivado High Level Synthesis (VHLS).
- HLS provides the digital level circuit from its behavioural description [1].

OPTIMIZATION DIRECTIVES

- **Loop unroll:** This is a compiler optimization technique that uses the parallelism by creating multiple copies of loop body.

  Directive command: #pragma HLS unroll factor =<INTEGER> [2].
- **Loop pipeline:** Optimization technique that allows, loop to be implemented in concurrent manner.

  Directive command: #pragma HLS Pipeline Initiation interval = <INTEGER> [2].


C Language

Apply directives
i. Loop unroll
ii. Loop pipeline

Assessment

Export &
Implemented

Fig.1 Basic block diagram of Design

CONCLUSION

Applications of optimization directives have explored to reduce execution time. Pipelining and unrolling show the reduction in delay by approx. 28% and 71%, and increase in hardware by 14% and 68% respectively, as compared to sequential. Simulation results show that proposed design has reduced time complexity for mathematical functions.

RESULTS

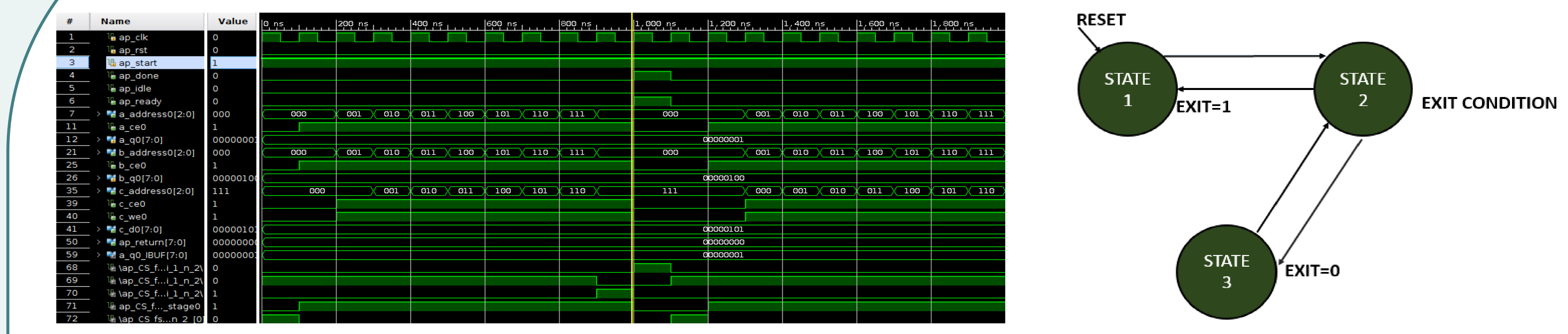


Fig.2 Simulation result and hardware realization of Sequential process

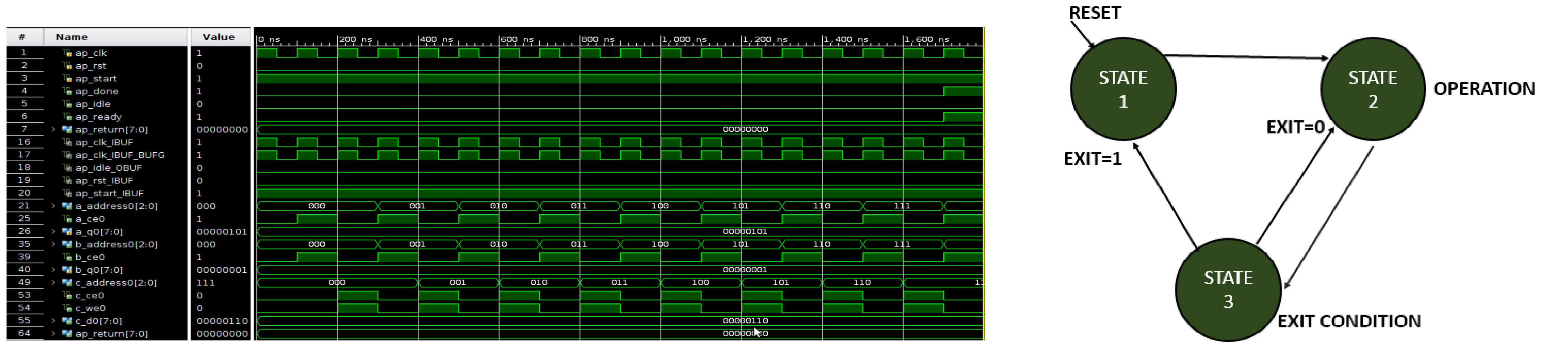


Fig.3 Simulation result and hardware realization after applying loop pipeline

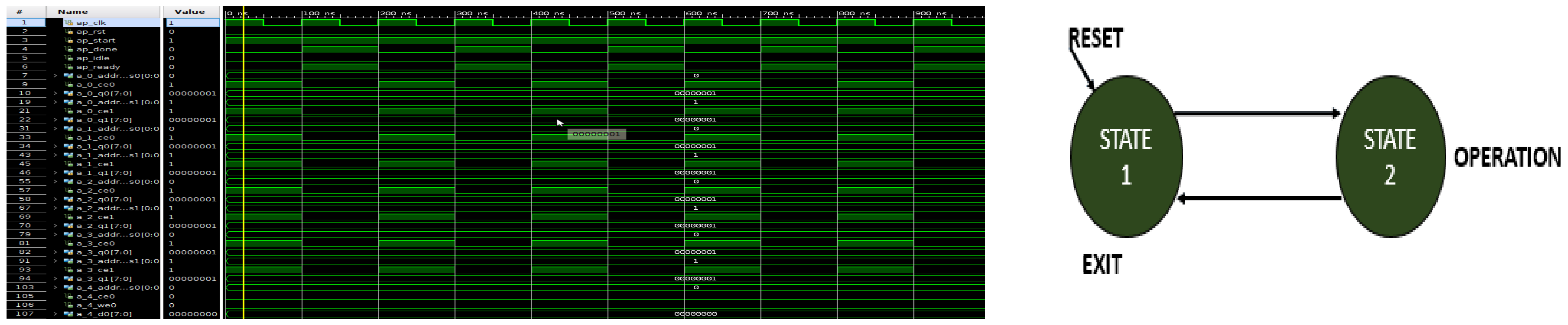


Fig.4 Simulation result and hardware realization after applying loop unrolling

PARAMETERS	Sequential	Pipeline	Unroll
Loop latency	17	10	1
LUTs	50	67	117
I/O ports	142	187	468

Table : Comparison of computational optimization techniques

REFERENCES

[1] Spyridon G., John E., *Overview of High-Level Synthesis tool*. Topical workshop on electronics for particle physics. 2010

[2] Sumit G., Rajesh G., Nikhil D. D., Alexandru N., *SPARK: A Parallelization approach to the High-Level Synthesis of Digital Circuit*. 2004, Springer Science US.

ACKNOWLEDGEMENT

I would like to convey my sincere thanks to Dr. Srivathsan Vasudevan and Dr. Satya S. Bulusu from IIT Indore for technical discussions.