

## Research Article

### Performance Analysis of Enhanced Adaptive Logic Module for High Performance FPGA Architecture

<sup>1</sup>V.J.K. Kishor Sonti, <sup>1</sup>Y. Varthamanan and <sup>2</sup>D. Sivaraman

<sup>1</sup>Department of E.C.E.,

<sup>2</sup>M. Tech VLSI Design, Sathyabama University, Chennai, Tamilnadu, India

**Abstract:** The Enhanced Adaptive Logic Module for High Performance Field Programmable Gate Array Architecture has been designed to increase the speed and reduces the device utilization of FPGA Architecture. The existing altera based adaptive logic module has eight input critical path look up tables. That means six inputs are given directly to the one Lookup table and two inputs are given to the other LUT by shorting any two from the first LUT of that six inputs. So that the delays has been increased. The proposed EALM has 8 fracturable direct input LUT, so eight input logic function can be implement by using this module with high speed than the altera based ALM. In this method only one LUT is used and it produces single output. Thus there is no need of carry logic, so that the area has been reduced and increases the speed. And FPGA architecture is formed by combining four adaptive logic modules using programmable interconnect array. Enhanced adaptive logic module is introduced in that architecture instead of adaptive logic module and the performance is analyzed. The adaptive logic modules and enhanced adaptive logic modules of field programmable gate array architecture has been designed and simulated by using Xilinx ISE 12.1.

**Keywords:** Adaptive logic module, enhanced adaptive logic module, field programmable gate array, lookup table, programmable interconnect array

## INTRODUCTION

The Adaptive Logic Module (Jason and Qiang, 2011) is the basic building block of logic in the Field Programmable Gate Array architecture (Farooq *et al.*, 2012). It provides advanced features with efficient logic usage and is designed to maximize the performance and resource usage. Field programmable Gate Arrays are pre-fabricated silicon devices that can be electrically programmed in the field to become almost any kind of digital circuitry system.

FPGAs provide cheaper solution and faster time to market as compared to Application Specific Integrated Circuits which normally require a lot of resources in terms of time and money to obtain first device. Field Programmable Gate Arrays were first introduced almost two and a half decades ago. Since then they have seen a rapid growth and have become a popular implementation media for digital circuits. The advancement in process technology has greatly enhanced the logic capacity of FPGAs and has in turn made them a viable implementation alternative for larger and complex designs. The basic FPGA architecture is as shown in Fig. 1. Normally FPGAs comprises of:

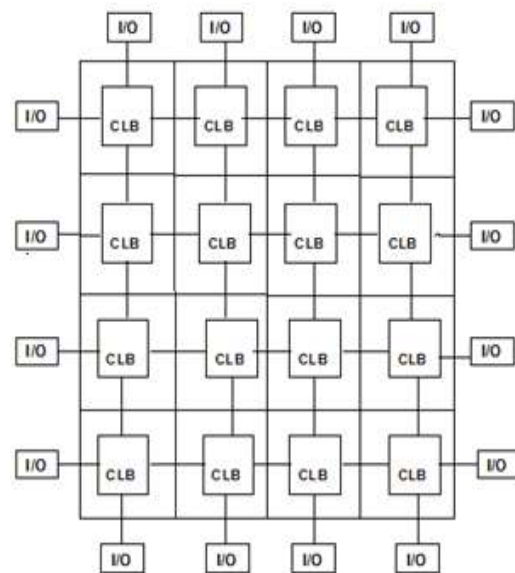


Fig. 1: Basic FPGA architecture

- Programmable logic blocks which implement logic functions.
- Programmable routing that connects these logic functions.

- I/O blocks that are connected to logic blocks through routing interconnect and that make off-chip connections.

The configurable logic blocks (Lakys *et al.*, 2012) are arranged in a two dimensional grid and are interconnected by programmable routing resources. I/O blocks are arranged at the periphery of the grid and they are also connected to the programmable routing interconnect. The “programmable/reconfigurable” term in FPGAs indicates their ability to implement a new function on the chip after its fabrication is complete. The reconfigurability programmability of an FPGA is based on an implementation alternative for larger and complex designs. Underlying programming technology (Farooq *et al.*, 2012), which can cause a change in behavior of a pre-fabricated chip after its fabrication.

### ADAPTIVE LOGIC MODULE

The basic building block of the Altera FPGA architecture provides advanced features with efficient logic utilization. Each ALM contains a variety of look-up table based resources that can be divided between two adaptive LUTs. With up to eight inputs to the two ALUTs, one ALM can implement various combinations of two functions. This adaptability allows the ALM to be completely backward-compatible with four input LUT architectures. One ALM can also implement any

function of up to six inputs and certain seven-input functions. The basic adaptive logic module architecture is as shown in the Fig. 2, which has the following elements in the architecture, 6-input LUT, 2-input LUT, Carry Logic, Multiplexer and Register.

In addition to the adaptive LUT based resources, each ALM contains one programmable register, dedicated full adders and Multiplexer. Through these dedicated resources, the ALM can efficiently implement various arithmetic functions and shift registers. Each ALM drives all types of interconnects: local, row, column, carry chain, shared arithmetic chain, register chain and direct link interconnects. The earliest FPGAs used 4 LUTs (David, 2013), established as the best LUT size to increase area efficiency. State of the art FPGAs are pointed towards speed. Interconnect in FPGAs is slow as compared with custom ASICs, due to the presence of programmable routing switches and the overheads enforced by programmability.

**Lookup tables:** Lookup Tables have been the backbone of FPGA logic blocks since the invention of FPGAs in the 1980s. A K-LUT, for example show in the Fig. 3 (Dugganapally *et al.*, 2008) is a one-output memory with K address lines that can implement any Boolean function that uses up to K variables. The earliest FPGAs used 4 LUTs, established as the best LUT size to increase area efficiency.

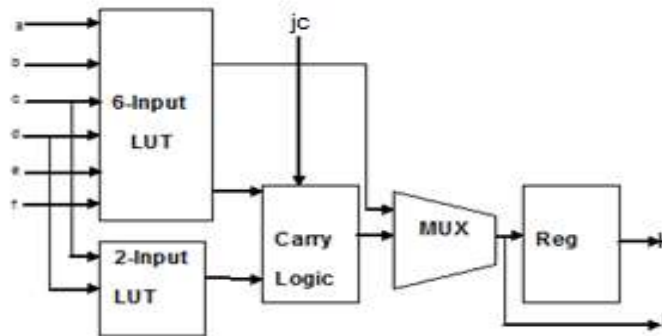


Fig. 2: ALM architecture

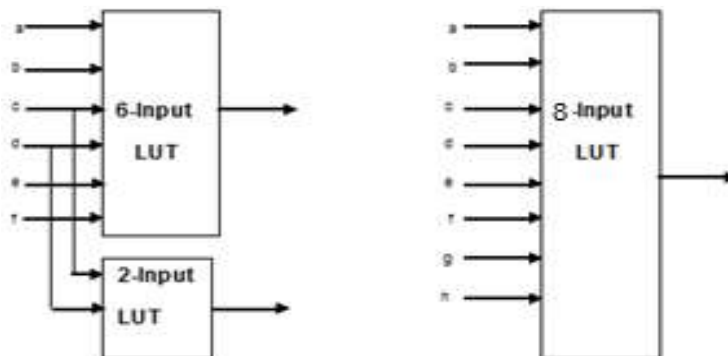


Fig. 3: Lookup tables

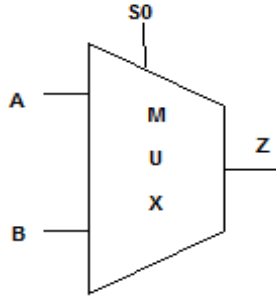


Fig. 4: Basic multiplexer

**Carry logic:** Carry logic is nothing but a full adder circuit. A full adder adds binary numbers and accounts for values carried in as well as out. The full-adder is usually a component in a cascade of adders, which add 8, 16, 32, etc., bit binary numbers. A full adder can be implemented in many different ways such as with a custom transistor-level circuit or composed of other gates (Chen *et al.*, 2006; Chong and Parameswaran, 2009).

**Multiplexer:** A multiplexer of  $2n$  inputs has  $n$  select lines, which are used to select which input line to send to the output. Multiplexers are mainly used to increase the amount of data that can be sent over the network within a certain amount of time and bandwidth. A multiplexer is also called a data selector (Chen *et al.*, 2006; Chong and Parameswaran, 2009). The basic multiplexer diagram is as shown in the Fig. 4.

An electronic multiplexer makes it possible for several signals to share one device or resource. Conversely, a demultiplexer is a device taking a single input signal and selecting one of many data-output-lines, which is connected to the single input. A

multiplexer is often used with a complementary demultiplexer on the receiving end.

**Register:** A shift register is a cascade of flip flops, sharing the same clock, in which the output of each flip-flop is connected to the data input of the next flip-flop in the chain, resulting in a circuit that shifts by one position the bit array stored in it, shifting in the data present at its input and shifting out the last bit in the array, at each transition of the clock input.

Shift registers can have both parallel and serial inputs and outputs. There are also bi-directional shift registers which allow shifting in both directions (Dugganapally *et al.*, 2008). The serial input and last output of a shift register can also be connected to create a circular shift register.

**Programmable interconnect array:** Programmable interconnect arrays are of interest to the semiconductor industry as a means of significantly reducing the cost associated with programmable interconnect arrays. The Programmable Interconnect Array can be defined as an array of potential interconnects that can be selectively switched from a high impedance state to a low impedance state. A typical programmable interconnect array is as shown in the Fig. 5. First, PIAs (Ian *et al.*, 2008) are of interest as a means of standardizing the packaging of integrated circuits. The programmable interconnect array.

Programmable Interconnect Array has two logic gates in the circuit namely AND Gate and OR Gate. There are four AND in PIA, each gate requires two inputs. One input is taken from the routing channel and the other input is EEPROM control, which is controlling the AND Gate. The four AND gate output is

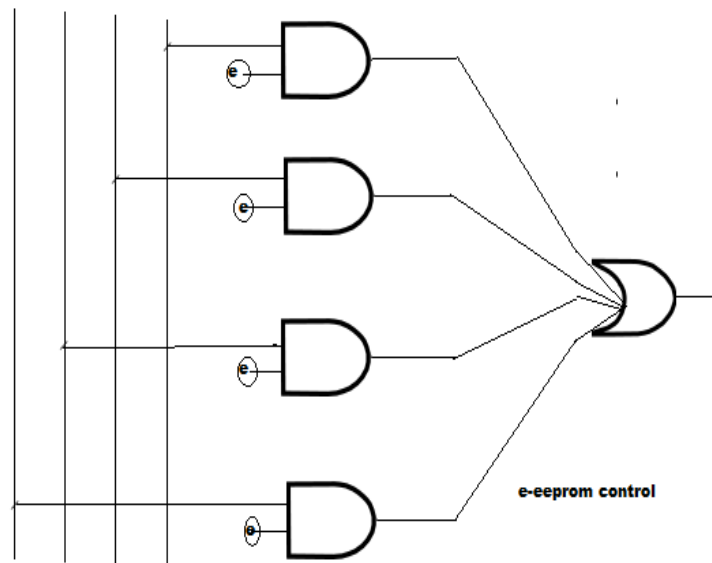


Fig. 5: Programmable interconnect array

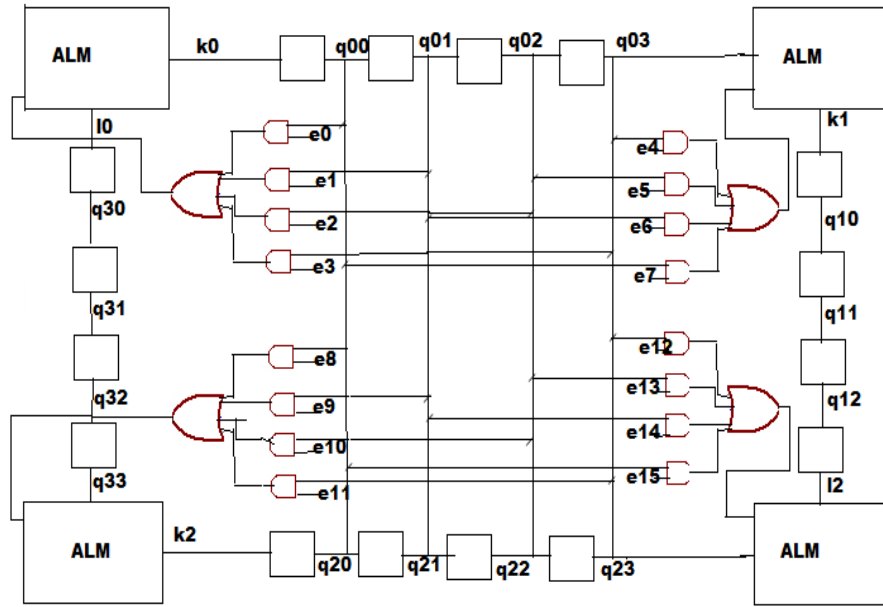


Fig. 6: ALM based FPGA architecture

fed into the four input OR gate. OR gate output is connected to the adaptive logic module.

**ALM based FPGA architecture:** The field programmable gate array can be designed by using four ALM (Ho *et al.*, 2009). The ALM as shown in the Fig. 6 is the basic building block of the Altera Stratix FPGA architecture, the adaptive logic module, provides advanced features with efficient logic utilization. Each ALM contains a variety of look-up table based resources that can be divided between two adaptive LUTs (David, 2013). With up to eight inputs to the two ALUTs, one ALM can implement various combinations of two functions. This adaptability allows the ALM to be completely backward-compatible with four input LUT architectures.

Four ALM would be connected by using the Programmable Interconnect Array. It is a programmable wiring path between I/O pins and ALMs and form one ALM to other ALMs. The PIA (Ian *et al.*, 2008) allows any I/O or ALM signal source to reach any destination on the architecture. Although it is fed by all macro cell and I/O block feedbacks, this fast, low skew PIA routes only those signal required implementing logic in each ALM.

### ENHANCED ADAPTIVE LOGIC MODULE

The earlier FPGAs (Ian *et al.*, 2008; Dugganapally *et al.*, 2008) used 4-LUTs, established as the best LUT size to increase area efficiency. Lookup Tables may be pre-calculated and stores the values in some places often it is taken and performs the operation. So it is

used to save more time. The LUT output is fed to flip flop and MUX as shown in the Fig. 7.

The existing adaptive logic module has two LUTs, so that each LUT gives one output. These outputs are fed into carry logic and get a output. But in proposed enhanced adaptive logic module has single LUT and single output, thus there is no need of carry logic. So that the device utilization of this enhanced adaptive logic module is less than the older adaptive logic module. Thus the area has been reduced, so again speed gets increased. Enhanced adaptive logic module has two inputs Multiplexer it takes two inputs and gives an output. The MUX output is the final output of this proposed EALM.

The proposed LUT based logic element has 8 fracturable direct inputs, so up to eight input logic function can implement by using this logic element with high speed than the altera FPGA adaptive logic module. It contains LUT, flip-flop and Multiplexer LUT is an array that reduces the computation time.

**EALM Based FPGA Architecture:** The field programmable gate array can be designed by using four enhanced adaptive logic module. The EALM is replaced in the ALM based architecture instead of ALM. It is the basic building block of the Altera Stratix FPGA architecture; the enhanced adaptive logic module provides advanced features with efficient logic utilization and high speed. One EALM can implement any function of up to eight-input functions.

These four EALM would be connected by using the Programmable Interconnect Array. There is a programmable wiring path between I/O pins and ALMs

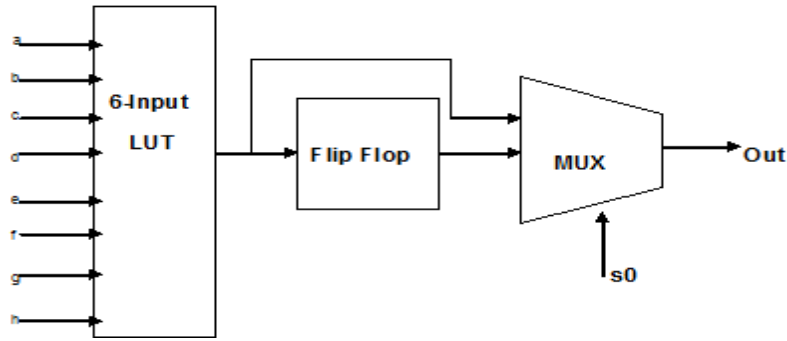


Fig. 7: EALM architecture

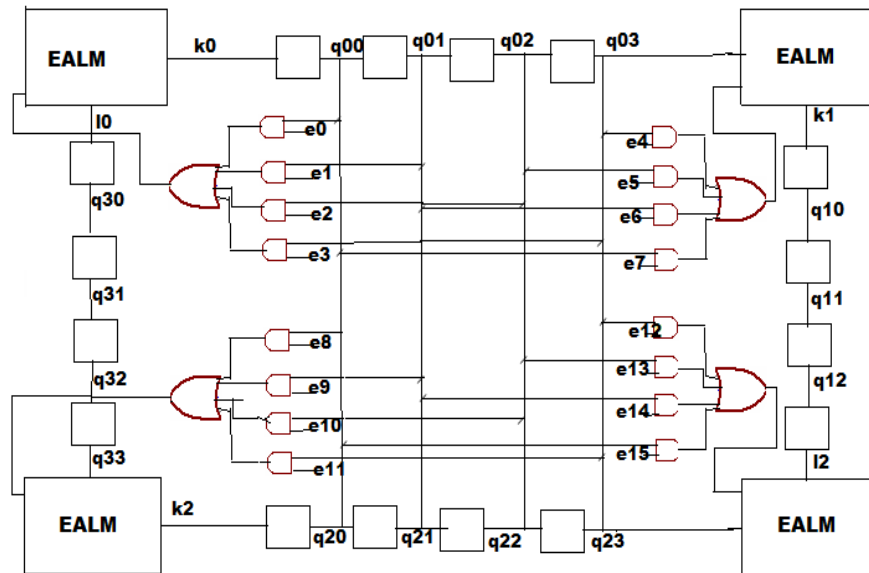


Fig. 8: EALM based FPGA architecture

and form one ALM to other ALMs. The PIA allows any I/O or ALM signal source to reach any destination on the architecture. Although it is fed by all macro cell and I/O block feedbacks, this fast, low-skew PIA routes only those signal required implementing logic in each ALM (Ian *et al.*, 2008).

The EALM based FPGA architecture is as shown in Fig. 8.

Enhanced adaptive logic module has two inputs Multiplexer it takes two inputs and gives a one output. The MUX (Chen *et al.*, 2006; Chong and Parameswaran, 2009) output is the final output of this proposed EALM. Altera FPGA architecture has shift register in it which contains four flip flops. The new LUT based FPGA logic element architecture has only one flip flop in it, so that the device utilization of this logic element is less than the older logic element. The field programmable gate array can be designed by using four Enhanced Adaptive Logic Module. The EALM is replaced in the ALM based architecture instead of ALM.

## RESULTS AND DISCUSSION

The output wave forms of adaptive logic module and enhanced adaptive logic modules are as shown and explained in the Fig. 9 and 10. The Proposed and Existing Logic elements device utilization is tabulated in Table 1 and the acceleration analysis is tabulated in Table 2. The ALMs have been simulated and measured their delays and device utilization by using Xilinx ISE 12.1 and the output is verified by using ModelSim.

The output waveform of adaptive logic module is as shown in the Fig. 9. Which has 6 direct (a, b, c, d, e, f) input and two (c1, d1) shorted input and two output from each LUT.

The output waveform of Enhanced adaptive logic module is as shown in the Fig. 10. Which has 8 direct (a, b, c, d, e, f, g, h) input and only one output k0.

The device utilization summary of ALM and EALM is tabulated in Table 1. This contains number of slices, number of 4-input look up tables, number of input outputs and input output buffers.

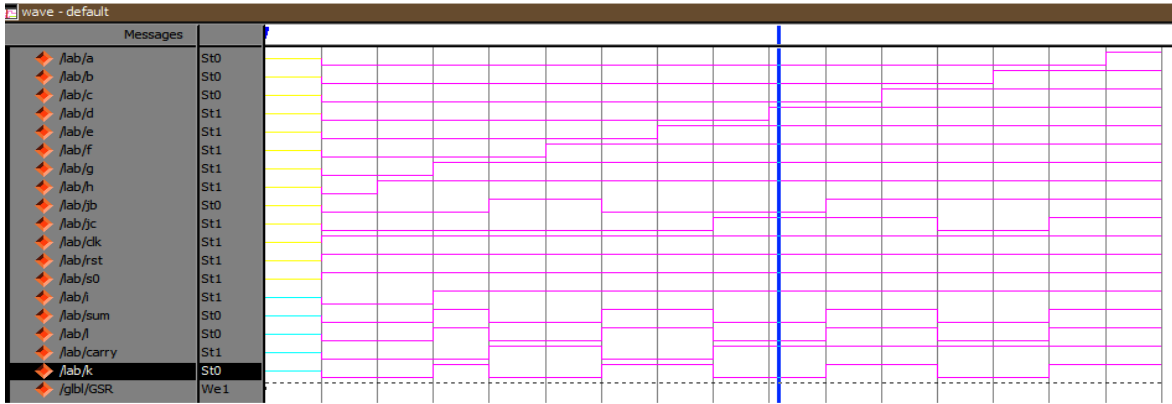


Fig. 9: ALM output waveform

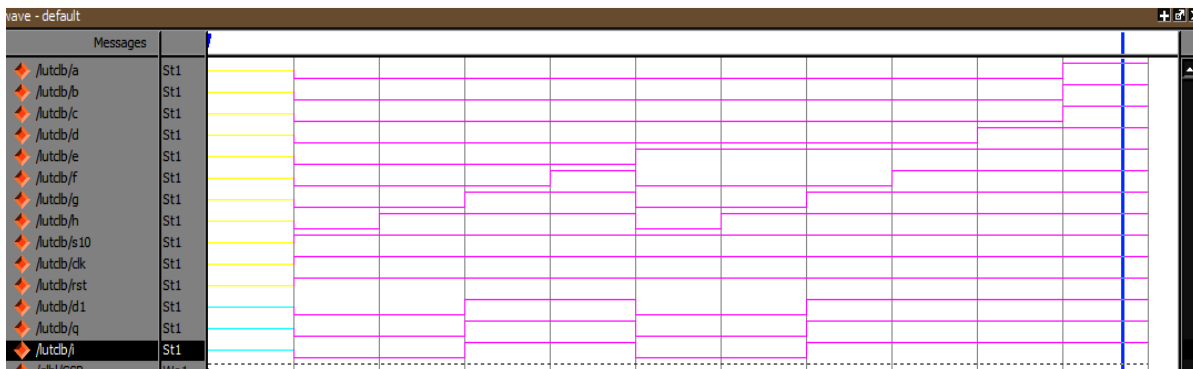


Fig. 10: EALM output waveform

Table 1: Device utilization of ALM and EALM

Devices	ALM	EALM
Number of Slices	3	2
Number of 4 input LUTs	6	3
Number of IOs	18	14
Number of bonded IOBs:	16	11

Table 2: Acceleration analysis of ALM and EALM

Delay	ALM	EALM
Net delay	IBUF: 0.849	IBUF: 1.218
	LUT4: 0.648	LUT4: 0.704
	LUT2: 0.648	LUT2: 0.704
	LUT4: 0.643	OBUF: 3.272
OBUF: 4.520		
Combinational path delay	9.519[ns]	7.619[ns]

Table 3: Device utilization summary existing and proposed methods

Devices	Existing	Proposed
Number of Slices	18	6
Number of 4 input LUTs	15	11
Number of IOs	84	68
Number of bonded IOs	72	52

Table 4: Acceleration analysis existing and proposed methods

Delay	Existing	Proposed
Net delay	IBUF: 0.849	IBUF: 0.849
	LUT4: 0.648	LUT4: 0.648
	LUT2: 0.648	LUT2: 0.648
	LUT4: 0.643	OBUF: 4.520
OBUF: 4.520		
Combinational path delay	11.621 [ns]	9.256 [ns]

The device utilization summary of ALM and EALM is tabulated in Table 2.

Figure 11 and 12 represent the ALM based and EALM based FPGA architectures respectively.

The device utilization summary of existing and proposed system is tabulated in Table 3.

The acceleration analysis of existing and proposed system is tabulated in Table 4. This contains net delay and combinational path delay.

### CONCLUSION

The proposed enhanced adaptive logic module packs more logic function compared to the existing ALM. To provides a highly efficient, high-performance FPGA. To provide a highly efficient, high-performance FPGA the designed EALMs are configured for the different outputs and were successfully simulated and verified to be functionally correct.

The enhanced adaptive logic module based field programmable gate array would be used in many digital circuits such as logic gates, adders and so on. This enhanced adaptive logic module will be used for many field programmable gate array logic applications. Also the results can be verified for further complex configurations.



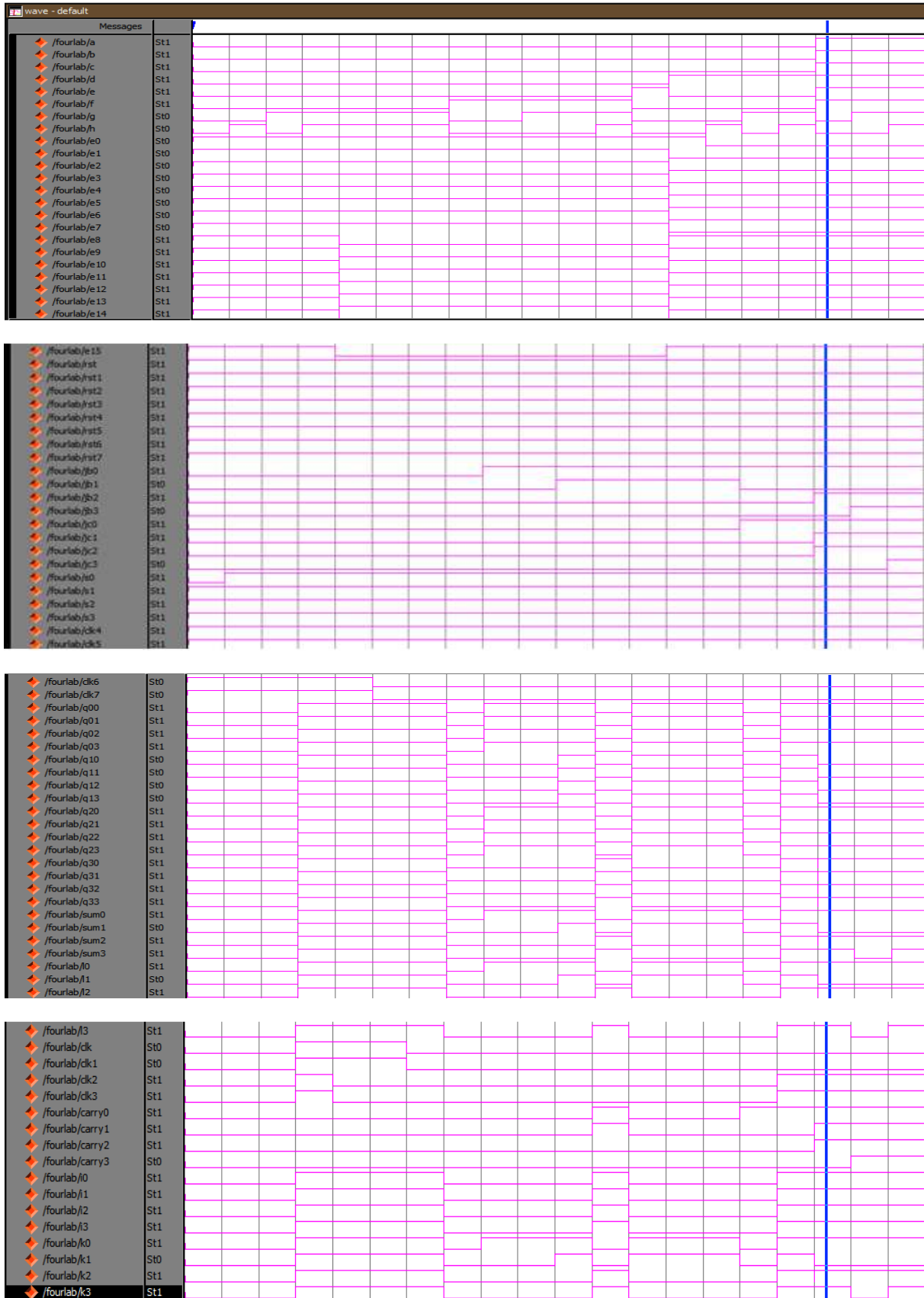


Fig. 11: ALM based FPGA architecture output waveform

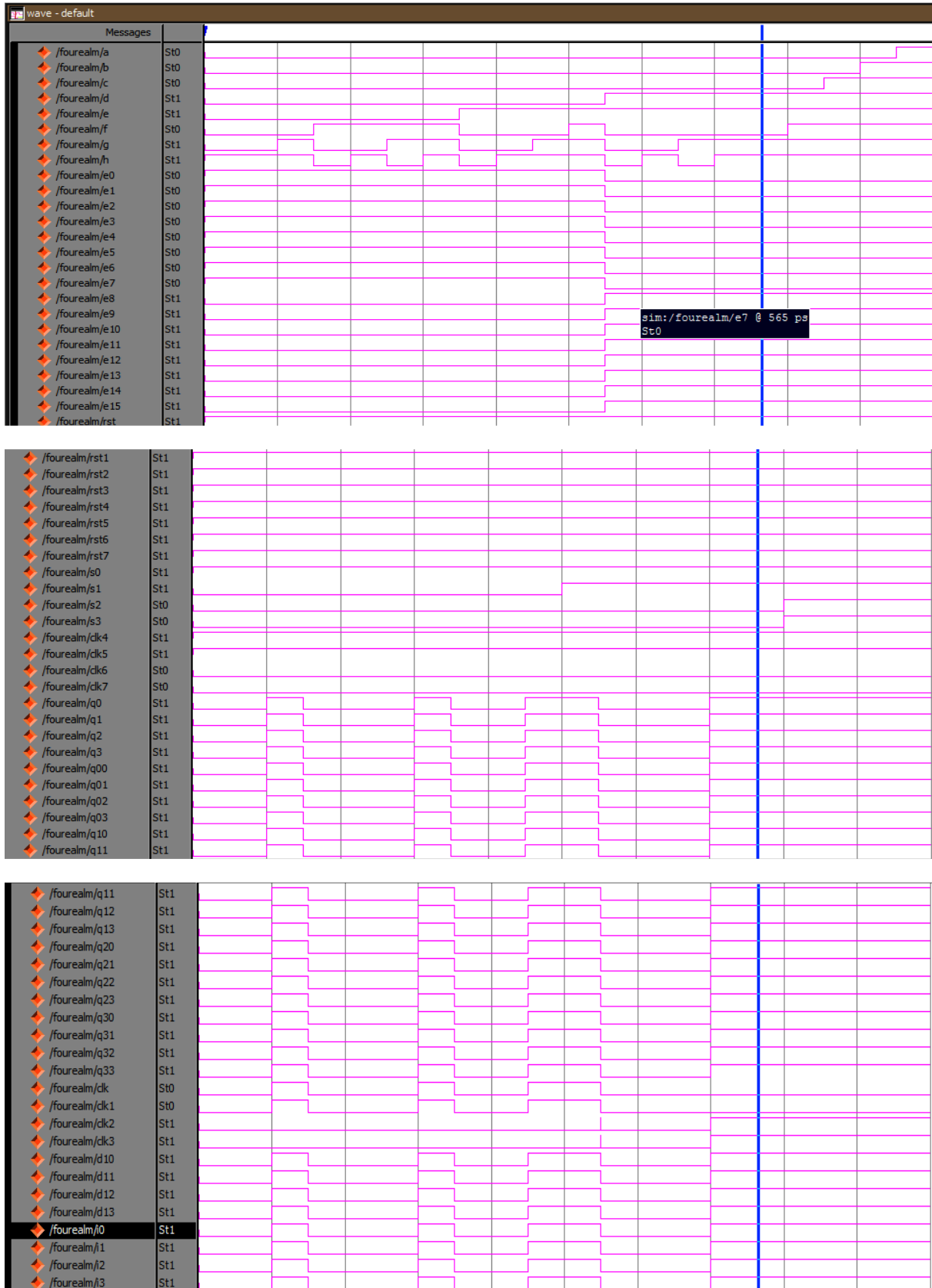


Fig. 12: EALM based FPGA architecture output waveform



## REFERENCES

- Chen, D., J. Cong and P. Pan, 2006. FPGA design automation: A survey. *Found. Trends Electr. Design Automat.*, 1(3): 195-330.
- Chong, Y.J. and S. Parameswaran, 2009. Flexible multi-mode embedded floating-point unit for field programmable gate arrays. *Proceeding of the ACM/SIGDA International Symposium on Field Programmable Gate Arrays (FPGA'09)*, pp: 171-180.
- David, B.T., 2013. The LUT-SR family of uniform random number generators for FPGA architectures. *IEEE T. VLSI Syst.*, 21(4).
- Dugganapally, I.P., W.K. Al-Assadi, T. Tammina and S. Smith, 2008. Design and implementation of FPGA configuration logic block using asynchronous static NCL. *Proceeding of the IEEE Region 5 Conference, Kansas City, MO*, pp: 1-6.
- Farooq, U., Z. Marrakchi and H. Mehrez, 2012. *Tree-based Heterogeneous FPGA Architectures: Application Specific Exploration and Optimization*. Springer, New York.
- Ho, C.H., C.W. Yu, P.H.W. Leong, W. Luk and S.J.E. Wilton, 2009. Floating-point FPGA: Architecture and modeling. *IEEE T. VLSI Syst.*, 17(2): 1709-1718.
- Ian, K., T. Russell and R. Jonathan, 2008. FPGA architecture: Survey and challenges. *Found. Trends Electr. Design Automat.*, 2(2): 135-253.
- Jason, H.A. and W. Qiang, 2011. Area-efficient FPGA logic elements: Architecture and synthesis. *Proceeding of the 16th Asia and South Pacific Design Automation Conference (ASP-DAC, 2011)*. Yokohama, pp: 369-375.
- Lakys, Y., W. Zhao, J. Klein and C. Chappert, 2012. MRAM crossbar based configurable logic block. *Proceeding of the IEEE International Symposium on, Circuits and Systems (ISCAS, 2012)*, pp: 2945-2948.