Hybrid Configurable Logic Architecture for FPGA's

[1]Mr. Y. Jeevan, [2]Mr.I.Sharath Chandra, [3]Prof. B. Kedarnath

[1,2]Assistant Professor, [3]Professor & HOD,
Department of ECE,
GNIT, Hyderabad, Telangana, India.

Abstract- **In this paper, we have proposed a Hybrid configurable logic block architecture for field-programmable gate arrays that contain a mixture of lookup tables and hardened multiplexers are evaluated toward the goal of higher logic density and area reduction. Technology mapping optimizations that target the proposed architectures are also implemented within ABC. Both accounting for complex logic block and routing area while maintaining mapping depth. For fracturable architectures, the proposed architecture of this paper analysis the logic size, area and power consumption using Xilinx 14.2.**

Index Terms: **FPGA, CLB's, Look Up Tables, Multiplexers, Fracturable Architectures.**

I. INTRODUCTION :

Hybrid configurable logic block architectures for field-programmable gate arrays that contain a mixture of lookup tables and hardened multiplexers are evaluated toward the goal of higher logic density and area reduction. Technology mapping optimizations that target the proposed architectures are also implemented within ABC. Both accounting for complex logic block and routing area while maintaining mapping depth. For fracturable architectures, the proposed architecture of this paper analysis the logic size, area and power consumption using Xilinx 14.2. Throughout the history of field-programmable gate arrays (FPGAs), lookup tables (LUTs) have been the primary logic element (LE) used to realize combinational logic. A *K*-input LUT is generic and very flexible—able to implement any *K*-input Boolean function. The use of LUTs simplifies technology mapping as the problem is reduced to a graph covering problem. However, an exponential area price is paid as larger LUTs are considered. The value of *K* between 4 and 6 is typically seen in industry and academia, and this range has been demonstrated to offer a good area/performance compromise. Recently, a number of other works have explored alternative FPGA LE architectures for performance improvement to close the large gap between FPGAs and application-specific integrated circuits (ASICs).The MUX-based logic blocks for the FPGAs have seen success in early commercial architectures, such as the Actel ACT-1/2/3 architectures, and efficient mapping to these structures has been studied in the early 1990s. However, their use in commercial chips has waned, perhaps partly due to the ease with which logic functions can be mapped into LUTs, simplifying the entire computer aided design (CAD) flow. Nevertheless, it is widely understood that the LUTs are inefficient at implementing MUXs, and that MUXs are frequently used in logic circuits. To underscore the inefficiency of LUTs implementing MUXs, consider that a six input LUT (6-LUT) is essentially a 64-to-1 MUX (to select 1 of 64 truth-table rows) and 64-SRAM configuration cells, yet it can only realize a 4-to-1 MUX (4 data + 2 select = 6 inputs).

II. OBJECTIVE :

Multiple hybrid configurable logic block architectures, both nonfracturable and fracturable with varying MUX:LUT logic element ratios are evaluated in yosys synthesis tool using a custom tool flow consisting of verilog coding, yosys front-end synthesis, ABC logic synthesis and technology mapping, and plan ahead Xilinx tool for packing, placement, routing, and architecture exploration. Technology mapping optimizations that target the proposed architectures are also implemented within ABC.
In this paper, we present a six-input LE based on a 4-to-1 MUX, MUX4, that can realize a subset of six-input Boolean logic functions, and a new hybrid complex logic block (CLB) that contains a mixture of MUX4s and 6-LUTs. The proposed MUX4s are small compared with a 6-LUT (15% of 6-LUT area), and can efficiently map all {2, 3}-input functions and some {4, 5, 6}-input functions. In addition, we explore fracturability of LEs—the ability to split the LEs into multiple smaller elements—in both LUTs and MUX4s to increase logic density. The ratio of LEs that should be LUTs versus MUX4s is also explored toward optimizing logic density for both nonfracturable and fracturable FPGA architectures.

III. EXISTING SYSTEM:

An algorithm for technology mapping of combinational and sequential logic networks is proposed and applied to mapping into K-input lookup-tables (K-LUTs). The new algorithm avoids the hurdle of computing all K-input cuts while preserving the quality of the results, in terms of area and depth. The memory and runtime of the proposed algorithm are linear in circuit size and quite affordable even for large industrial designs. For example, computing a good quality 6-LUT mapping of an AIG with 1M nodes takes 150Mb of RAM and 1 minute on a typical laptop. An extension of the algorithm allows for sequential mapping, which searches the combined space of all possible mappings and retimings. This leads to an 18-22% improvement in depth with a 3-5% LUT count penalty, compared to combinational mapping followed by retiming.

IV. PROPOSED SYSTEM

MUX4: 4-to-1 Multiplexer Logic Element

The MUX4 LE shown in Fig. 1 consists of a 4-to-1 MUX with optional inversion on its inputs that allow the realization of any {2, 3}-input function, some {4, 5}-input functions, and one 6-input function—a 4-to-1 MUX itself with optional inversion on the data inputs. A 4-to-1 MUX matches the input pin count of a 6-LUT, allowing for fair comparisons with respect to the connectivity and intra-cluster routing.

Naturally, any two-input Boolean function can be easily implemented in the MUX4: the two function inputs can be tied to the select lines and the truth table values (logic-0 or logic-1) can be routed to the data inputs accordingly. Or alternately, a Shannon decomposition can be performed about one of the two variables—the variable can then feed a select input. The Shannon cofactors will contain at most one variable and can, therefore, be fed to the data inputs (the optional inversion may be needed).

Logic Elements, Fracturability, and MUX4-Based Variants

Two families of architectures were created: 1) without fracturable LEs and 2) with fracturable LEs. In this paper, the fracturable LEs refer to an architectural element on which one or more logic functions can be optionally mapped. Nonfracturable LEs refer to an architectural element on which only one logic function is mapped. In the nonfracturable architectures, the MUX4 element is used together with nonfracturable 6-LUTs. This element shares the same number of inputs as a 6-LUT lending for fair comparison with respect to the input connectivity.

Hybrid Complex Logic Block:

A variety of different architectures were considered—the first being a nonfracturable architecture. In the nonfracturable architecture, the CLB has 40 inputs and ten basic LEs (BLEs), with each BLE having six inputs and one output following empirical data in prior work as in nonfracturable CLB architecture with BLEs that contain an optional register.

We vary the ratio of MUX4s to LUTs within the ten element CLB from 1:9 to 5:5 MUX4s:6-LUTs. The MUX4 element is proposed to work in conjunction with 6-LUTs, creating a hybrid CLB with a mixture of 6-LUTs and MUX4s (or MUX4 variants). Fig. 2 shows the organization of our CLB and internal BLEs.

MUX4:

The MUX4 LE shown in Fig. 1 consists of a 4-to-1 MUX with optional inversion on its inputs that allow the realization of any {2, 3}-input function, some {4, 5}-input functions, and one 6-input function—a 4-to-1 MUX itself with optional inversion on the data inputs. A 4-to-1 MUX matches the input pin count of a 6-LUT, allowing for fair comparisons with respect to the connectivity and intra-cluster routing. Naturally, any two-input Boolean function can be easily implemented in the MUX4: the two function inputs can be tied to the select lines and the truth table values (logic-0 or logic-1) can be routed to the data inputs accordingly. Or alternately, a Shannon decomposition can be performed about one of the two variables—the variable can then feed a select input. The Shannon cofactors will contain at most one variable and can, therefore, be fed to the data inputs (the optional inversion may be needed). For three-input functions, consider that a Shannon decomposition about one variable produces cofactors with at most two variables. A second decomposition of the cofactors about one of their two remaining variables produces cofactors with at most one variable. Such single-variable cofactors can be fed to the data inputs (the optional inversion may be needed), with the decomposition variables feeding the select inputs. Likewise, functions of more than four inputs can be implemented in the MUX4 as long as Shannon decomposition with respect to any two inputs produce cofactors with at most one input. Observe that input inversion on each select input is omitted as this would only serve to permute the four MUX data inputs. While this could help routability within the CLB's internal crossbar, additional inversions on the select inputs would not increase the number of Boolean functions that are able to map to the MUX4 LE.

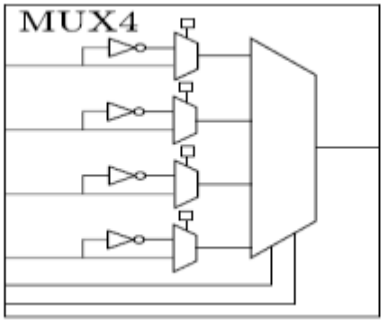

Fig 1: MUX4 LE

DUAL MUX:

For the MUX4 variant, Dual MUX4, we use two MUX4s within a single eight-input LE. In the configuration, shown in Fig.2, the two MUX4s are wired to have dedicated select inputs and shared data inputs. This configuration allows this structure to map two independent (no shared inputs) three-input functions, while larger functions may be mapped dependent on the shared inputs between both functions

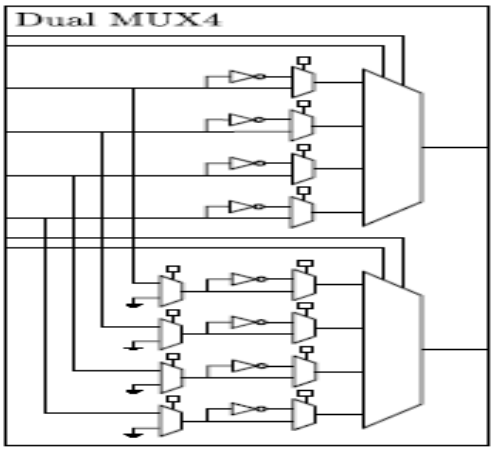

Fig 2: Dual MUX4

MODULE'S AND THEIR DESCRIPTION:

- INVERTER
- 2:1 MUX
- 4:1 MUX
- D-FLIPFLOP

INVERTER:

A NOT gate (also often called Inverter) is a logic gate. Each NOT gate has only one input signal. Logically with NOT gates, the input and the output swap, so if you input *1* it outputs as *0*; likewise if you input *0* it outputs as *1*. Generally, actual NOT gates with a voltage below 0.5V is *0*, and a voltage of 4–5V is *1*.

Input	Output
1	0
0	1

Fig 3: INVERTER

2:1 MUX:

A two–to–one multiplexer is a combinational circuit that uses one control switch (S) to connect one of two input data lines (D1 or D0) to a single output (F). Only one of the input data lines can be aligned to the output of the multiplexer at any given time. It's like sharing ice–cream on a date with one spoon. At any given instant your hand is either feeding you or your partner with the spoon — but not both mouths at the same time (Unless you have some creative ideas for sharing ice–cream on a date).Fig 4 below shows the block diagram symbol of the two–to–one multiplexer. The wedged shape is supposed to depict how the circuit funnels one of two inputs to a single output. We also show the truth table of the 2x1 mux in Table 1. The table shows how the selector switch controls which input line feeds the output. When S = 0, F = D0; when S =1, F = D1.

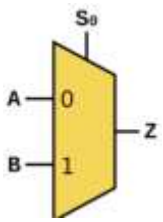

Fig 4 : 2:1MULTIPLEXER

Table 1: 2:1 MUX TRUTH TABLE

S	F
0	D0
1	D1

The 4–bit parallel–access shift register illustrates a straight forward application of the 2x1 mux. In that application the 2x1 mux allows a register to operate in two possible modes: parallel or serial. When the selector switch is zero (S=0) the register is in serial mode; when the switch is one (S=1) the register is in parallel mode.

4:1 MUX:

Multiplexers operate like very fast acting multiple position rotary switches connecting or controlling multiple input lines called "channels" one at a time to the output. Multiplexers, or MUX's, can be either digital circuits made from high speed logic gates used to switch digital or binary data or they can be analogue types using transistors, MOSFET's or relays to switch; one of the voltage or current inputs through to a single output.

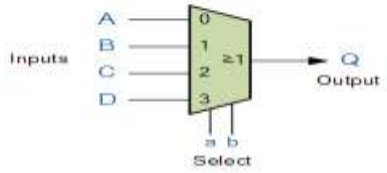

Fig.5: 4:1 MULTIPLEXER

D-FLIPFLOP

One of the main disadvantages of the basic SR NAND Gate bistable circuit is that the indeterminate input condition of SET = "0" and RESET = "0" is forbidden. This state will force both outputs to be at logic "1", over-riding the feedback latching action and whichever input goes to logic level "1" first will lose control, while the other input still at logic "0" controls the resulting state of the latch. But in order to prevent this from happening an inverter can be connected between the "SET" and the "RESET" inputs to produce another type of flip flop circuit known as a Data Latch**,** Delay flip flop**,** D-type Bistable**,** D-type Flip Flop or just simply a D Flip Flop as i*t is more generally called. The D Flip Flop **is** by far the most important of the clocked flip-flops as it ensures that ensures that inputs S and R are never equal to one at the same time. The D-type flip flop are constructed from a gated SR flip-flop with an inverter added between the S and the R inputs to allow for a single D (data) input. Then this single data input, labelled D, is used in place of the "set" signal, and the inverter is used to generate the complementary "reset" input thereby making a level-sensitive D-type flip-flop from a level-sensitive RS-latch as now S = D and R = not D as shown

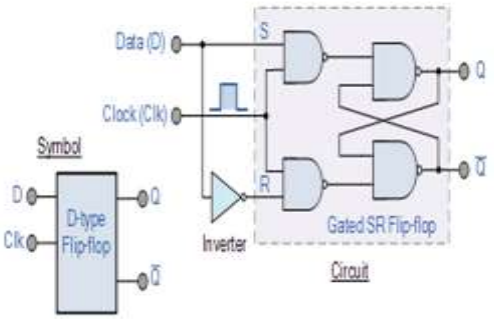

Fig.6: D-TYPE FLIPFLOP CIRCUIT

The "D flip flop" will store and output whatever logic level is applied to its data terminal so long as the clock input is HIGH. Once the clock input goes LOW the "set" and "reset" inputs of the flip-flop are both held at logic level "1" so it will not change state and store whatever data was present on its output before the clock transition occurred. In other words the output is "latched" at either logic "0" or logic "1".

Table 2:Truth Table for D flip flop

Clk	D	Q		Description
↓ » 0	X	Q	$\overline{Q}$	Memory no change
↑ » 1	0	0	1	Reset Q » 0
↑ » 1	1	1	0	Set Q » 1

A variety of different architectures were considered—the first being a nonfracturable architecture. In the nonfracturable architecture, the CLB has 40 inputs and ten basic LEs (BLEs), with each BLE having six inputs and one output following empirical data in prior work. Fig. 2 shows this nonfracturable CLB architecture with BLEs that contain an optional register. We vary the ratio of MUX4s to LUTs within the ten element CLB from 1:9 to 5:5 MUX4s:6-LUTs. The MUX4 element is proposed to work in conjunction with 6-LUTs, creating a hybrid CLB with a mixture of 6-LUTs and MUX4s (or MUX4 variants). Fig. 2 shows the organization of our CLB and internal BLEs.

Logic Elements, Fracturability, and MUX4-Based Variants Two families of architectures were created: 1) without fracturable LEs and 2) with fracturable LEs. In this paper, the fracturable LEs refer to an architectural element on which one or more logic functions can be optionally mapped. Nonfracturable LEs refer to an architectural element on which only one logic function is mapped. In the nonfracturable architectures, the MUX4 element shown in Fig. 7 is used together with nonfracturable 6-LUTs.

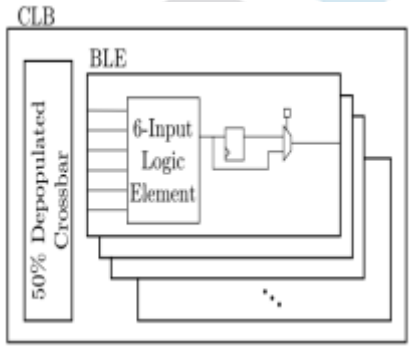

Fig 7: NON FRACTURABLE HYBRID CLB

This element shares the same number of inputs as a 6-LUT lending for fair comparison with respect to the input connectivity. For fracturable architectures, the CLB has 80 inputs and ten BLEs, with each BLE having eight inputs and two outputs emulating an Altera Stratix Adaptive-LUT. The same sweep of MUX4 to LUT ratios was also performed. Fig.7 shows the fracturable architecture with eight inputs to each BLE that contains two optional registers.Basic LUT consists of six inputs. As we are replacing LUT with a multiplexer. The MUX4 is treated as six input LUT in which the four data lines and two selection lines are considered as 6 input lines.The logic in the CLB is implemented by using MUX4.

MUX4:

The MUX4 LE shown in Fig. 1 consists of a 4-to-1 MUX with optional inversion on its inputs that allow the realization of any {2, 3}-input function, some {4, 5}-input functions, and one 6-input function—a 4-to-1 MUX itself with optional inversion on the data inputs. A 4-to-1 MUX matches the input pin count of a 6-LUT, allowing for fair comparisons with respect to the connectivity and intra-cluster routing. Naturally, any two-input Boolean function can be easily implemented in the MUX4: the two function inputs can be tied to the select lines and the truth table values (logic-0 or logic-1) can be routed to the data inputs accordingly. Or alternately, a Shannon decomposition can be performed about one of the two variables—the variable can then feed a select input. The Shannon cofactors will contain at most one variable and can, therefore, be fed to the data inputs (the optional inversion may be needed). For three-input functions, consider that a Shannon decomposition about one variable produces cofactors with at most two variables.

A second decomposition of the cofactors about one of their two remaining variables produces cofactors with at most one variable. Such single-variable cofactors can be fed to the data inputs (the optional inversion may be needed), with the decomposition variables feeding the select inputs. Likewise, functions of more than four inputs can be implemented in the MUX4 as long as Shannon decomposition with respect to any two inputs produce cofactors with at most one input. Observe that input inversion on each select input is omitted as this would only serve to permute the four MUX data inputs. While this could help routability within the CLB's internal crossbar, additional inversions on the select inputs would not increase the number of Boolean functions that are able to map to the MUX4 LE.

Working process of MUX4:
The four inputs a,b,c,d are given to the 4:1 MUX where the individual inputs are given to the 2:1 MUX with one input compliment and the other input is non-compliment and also with two selection lines.

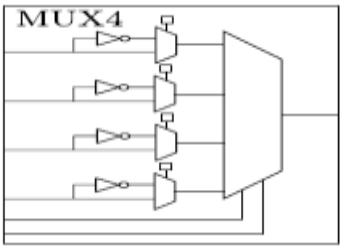

Fig 8: MUX4

Operation of MUX4:

- The operation of MUX4 is explained clearly by taking the four inputs a,b,c,d as 1001 with selection line s [3:0] as a high inputs 1111.
- The hybrid output is being obtained by selecting the selection lines sel [1:0] as 00,01,10,11.
- The inputs are taken as a vector e [3:0].when sel[1:0] is given as 01 then the hybrid_out is 0, where e[1] is being selected as output.

DUAL MUX:

For the MUX4 variant, Dual MUX4, we use two MUX4s within a single eight-input LE. In the configuration, shown in Fig.8, the two MUX4s are wired to have dedicated select inputs and shared data inputs. This configuration allows this structure to map two independent (no shared inputs) three-input functions, while larger functions may be mapped dependent on the shared inputs between both functions.

Working process of Dual MUX4:

As we are using a Dual MUX4, the four inputs are taken as a vector A[3:0] and thee are being shared between two MUX4 named M1 and M2 respectively. The inputs as a vector A[3:0] are given as complimented and uncomplimented inputs to four 2:1 MUX'S individually which gives outputs e,f,g,h respectively The input s is given as selection line for 2:1 multiplexers. These outputs are in turn given to the M1 MUX4 as inputs. The same inputs as a vector A[3:0] are given to four 2:1 MUXs where one input is shared input and the other inputs are grounded individually which gives i,j,k,l as outputs. These outputs are given as complimented and uncomplimented inputs to second set of 2:1 MUXs yielding as s,t,u,v outputs. Where these outputs are given to M2 MUX4 as inputs. The inputs s1 and s2 are given as selection lines for the first and second set of 2:1 multiplexers respectively. The inputs sel [1:0] is given as a selection line to dual MUX4.

Operation of Dual MUX4:

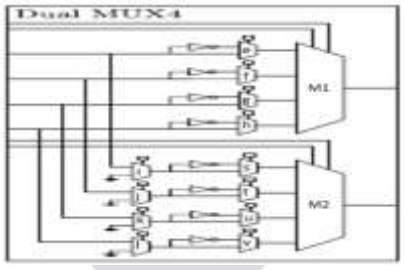

Fig 9: Dual MUX 4

- When the inputs as a vector A[3:0] are given as 0110 with selection line s [3:0] as a high inputs 1111.The inputs of S1[3:0], S2[3:0],SEL[3:0] are given as 0000.
- When the input for SEL1[1:0] is given as 11 and SEL2[1:0] is given as 11 and the reset value is set to 0 and the clock value for leading edge value is given as 1 and for trailing edge value is given as 0.

The outputs for dual MUX4 are

HYBRID_OUT : 0
HYBRID_OUT1 : 1

HYBRID CLB:

A variety of different architectures were considered—the first being a nonfracturable architecture. In the nonfracturable architecture, the CLB has 40 inputs and ten basic LEs (BLEs), with each BLE having six inputs and one output following empirical data in prior work. Fig. 7 shows this nonfracturable CLB architecture with BLEs that contain an optional register. We vary the ratio of MUX4s to LUTs within the ten element CLB from 1:9 to 5:5 MUX4s:6-LUTs. The MUX4 element is proposed to work in conjunction with 6-LUTs, creating a hybrid CLB with a mixture of 6-LUTs and MUX4s (or MUX4 variants). Fig. 7 shows the organization of our CLB and internal BLEs.

Logic Elements, Fracturability, and MUX4-Based Variants Two families of architectures were created: 1) without fracturable LEs and 2) with fracturable LEs. In this paper, the fracturable LEs refer to an architectural element on which one or more logic functions can be optionally mapped. Nonfracturable LEs refer to an architectural element on which only one logic function is mapped. In the nonfracturable architectures, the MUX4 element is used together with nonfracturable 6-LUTs.

This element shares the same number of inputs as a 6-LUT lending for fair comparison with respect to the input connectivity. For fracturable architectures, the CLB has 80 inputs and ten BLEs, with each BLE having eight inputs and two outputs emulating an Altera Stratix Adaptive-LUT.

The same sweep of MUX4 to LUT ratios was also performed and the fracturable architecture with eight inputs to each BLE that contains two optional registers.

Working process of Hybrid CLB:

The hybrid CLB consists of Dual MUX4 D-flipflop and 2:1 MUXs are placed as shown in below figure.

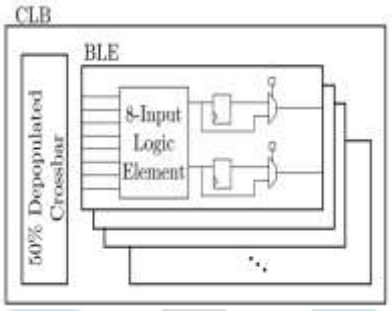

Fig 10: Hybrid CLB

The outputs of M1 and M2 are given to D-flipflop for creating some delay. The delayed and undelayed outputs are given to 2:1 multiplexers respectively. The outputs of the MUX4 are represented as HYBRID_OUT and HYBRID_OUT1 respectively. The outputs from MUX4 are given to D- flipflop yielding as clb1 and clb2 outputs respectively. The outputs of D-flipflop are given to two 2:1 MUXs which inturn gives us clbout1 and clbout2 as outputs respectively. These outputs clbout1 and clbout2 are proposed outputs of hybrid CLB.

Operation of Hybrid CLB:

- When the inputs as a vector A[3:0] are given as 0110 with selection line s [3:0] as a high inputs 1111.The inputs of S1[3:0], S2[3:0],SEL[3:0] are given as 0000.
- When the input for SEL1[1:0] is given as 11 and SEL2[1:0] is given as 11 and the reset value is set to 0 and the clock value for leading edge value is given as 1 and for trailing edge value is given as 0.

The outputs for dual MUX4 are

Clbout1 : 1
Clbout2 : 0
HYBRID_OUT : 0
HYBRID_OUT1 : 1

8.2 Output Waveforms of Components:

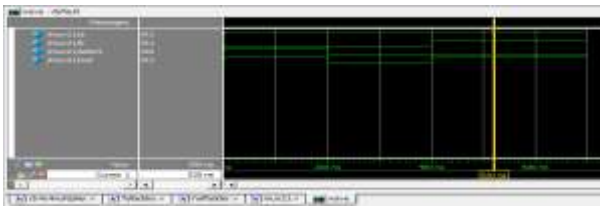

Fig22: output waveform of 2:1 multiplexer

Fig8.2: output waveform of D-flipflop

Fig23: output waveform of MUX4

Fig24: output waveform of Hybrid CLB

DEVICE UTILIZATION OF MUX4:

Slice Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slice LUTs	2	5,720	1%
Number used as logic	2	5,720	1%
Number used as Memory	0	1,440	0%
Number of occupied Slices	1	1,430	1%
Average Fanout of Non-Clock Nets	1.09		

DEVICE UTILIZATION OF HYBRID CLB:

Slice Logic Utilization	Used	Available	Utilization
Number of Slice Registers	2	11,440	1%
Number of Slice LUTs	8	5,720	1%
Number used as logic	8	5,720	1%
Number used as Memory	0	1,440	0%
Number of occupied Slices	3	1,430	1%
Number with an unused Flip Flop	6	8	75%
Number with an unused LUT	0	8	0%
Number of fully used LUT-FF pairs	2	8	25%
Number of unique control sets	1		
Number of slice register sites lost to control set restrictions	6	11,440	1%
Average Fanout of Non-Clock Nets	1.34		

RTL SCHEMATIC

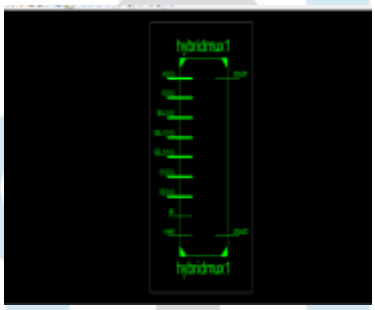

Fig25:RTL Schematic of hybridmux

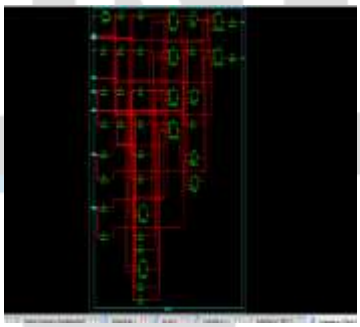

Fig26: Technology Scematic of hybridmux

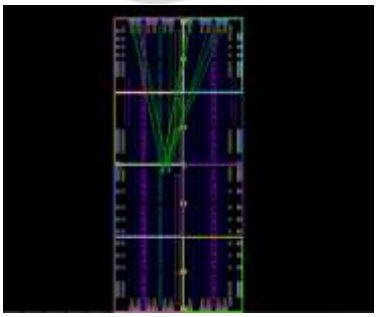

Fig27: Place and Routing

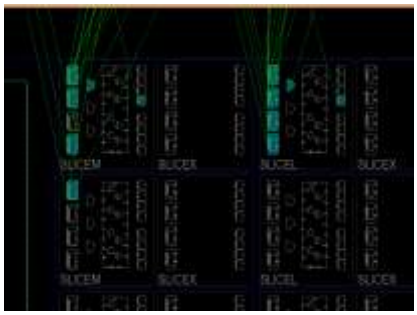

Fig28: Internal view of Place and Routing

9.3 CONCLUSION

We have proposed a new hybrid CLB architecture containing MUX4 hard MUX elements and shown techniques for efficiently mapping to these architectures. new multiplexer based truncation scheme with lower average and mean square errors than existing methods. For both non fracturable and fracturable architectures, we see minimal impact on timing performance for the architectures with best area-efficiency. the addition of MUX4s to FPGA architectures minimally impact FMax and show potential for improving logic-density in nonfracturable architectures and modest potential for improving logicdensity in fracturable architectures.

References:

[1] J. Rose *et al.*, "The VTR project: Architecture and CAD for FPGAs from verilog to routing," in *Proc. ACM/SIGDA FPGA*, 2012, pp. 77–86.
[2] Y. Hara, H. Tomiyama, S. Honda, and H. Takada, "Proposal and quantitative analysis of the CHStone benchmark program suite for practical C-based high-level synthesis," *J. Inf. Process.*, vol. 17,pp. 242–254, Oct. 2009.
[3] A. Canis *et al.*, "LegUp: High-level synthesis for FPGA-based processor/accelerator systems," in *Proc. ACM/SIGDA FPGA*, 2011,pp. 33–36.
[4] E. Ahmed and J. Rose, "The effect of LUT and cluster size on deepsubmicron FPGA performance and density," *IEEE Trans. Very Large Scale Integr. (VLSI)*, vol. 12, no. 3, pp. 288–298, Mar. 2004.
[5] J. Rose, R. Francis, D. Lewis, and P. Chow, "Architecture of fieldprogrammable gate arrays: The effect of logic block functionality on area efficiency," *IEEE J. Solid-State Circuits*, vol. 25, no. 5,pp. 1217–1225, Oct. 19 90.