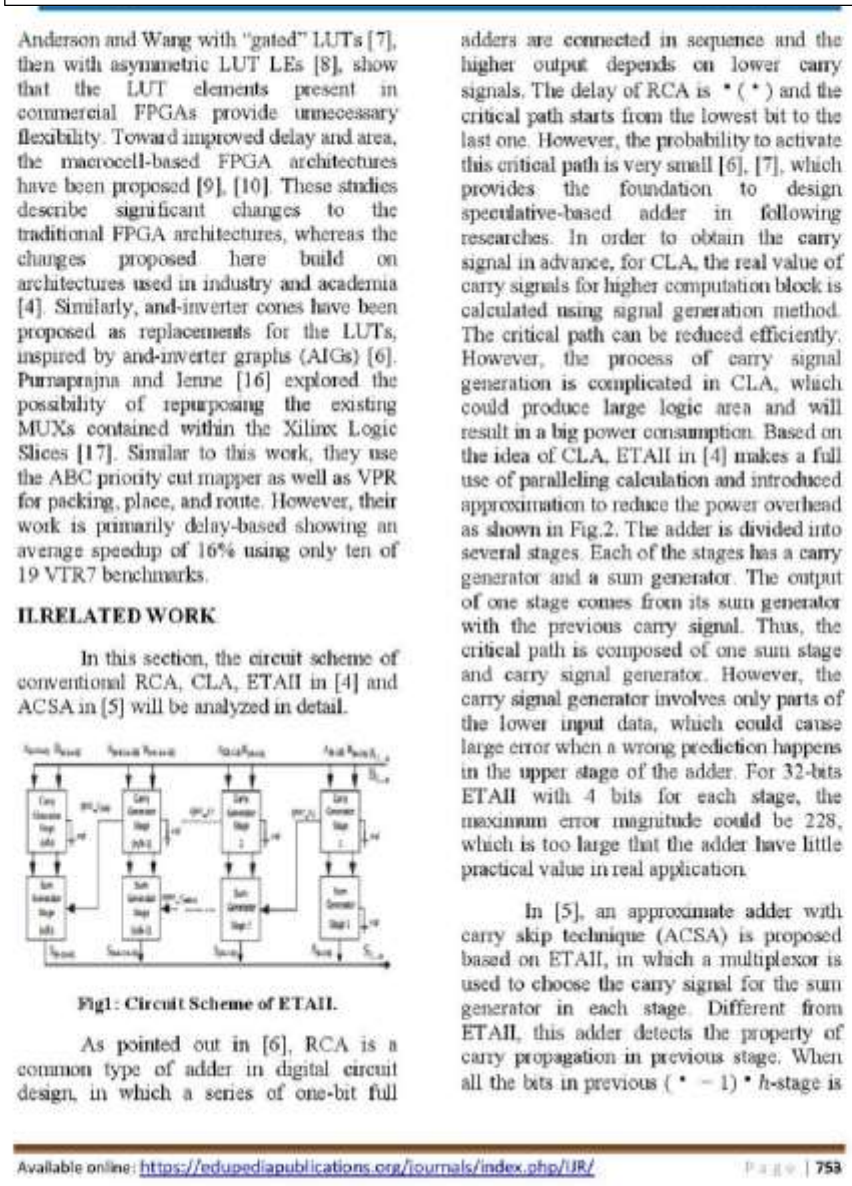
Design And Implementation of a Hybrid LUT/Multiplexer Architectures for FPGA

[1]R V Simhadri, [2]K.Devi Sri Snigdha,[3]S.Vaishnavi
[1] Assistant Professor, Dept. Of ECE, Pragati engineering College, adb road, surampalem, A.P, India.
[2]Assistant Professor, Dept. Of ECE, Pragati engineering College, adb road, surampalem, A.P, India.
[3]Assistant Professor, Dept. Of ECE, Pragati engineering College, adb road, surampalem, A.P, India.



Anderson and Wang with "gated" LUTs [7], then with asymmetric LUT LEs [8], show that the LUT elements present in commercial FPGAs provide unnecessary flexibility. Toward improved delay and area, the macrocell-based FPGA architectures have been proposed [9], [10]. These studies describe significant changes to the traditional FPGA architectures, whereas the changes proposed here build on architectures used in industry and academia [4]. Similarly, and-inverter cones have been proposed as replacements for the LUTs, inspired by and-inverter graphs (AIGs) [6]. Purnaprajna and Ienne [16] explored the possibility of repurposing the existing MUXs contained within the Xilinx Logic Slices [17]. Similar to this work, they use the ABC priority cut mapper as well as VPR for packing, place, and route. However, their work is primarily delay-based showing an average speedup of 16% using only ten of 19 VTR7 benchmarks.

II.RELATED WORK

In this section, the circuit scheme of conventional RCA, CLA, ETAII in [4] and ACSA in [5] will be analyzed in detail.

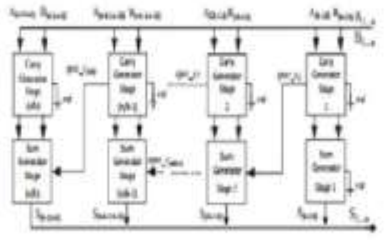
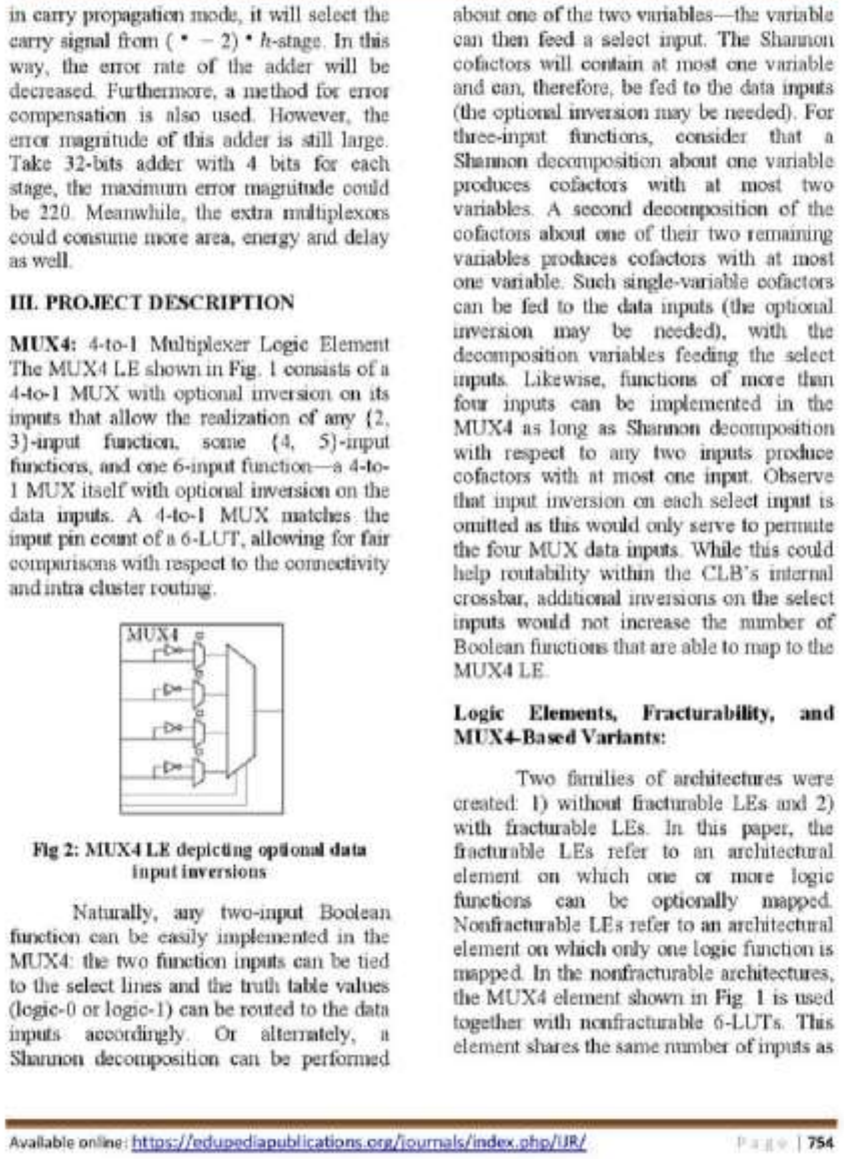


Fig1: Circuit Scheme of ETAII.

As pointed out in [6], RCA is a common type of adder in digital circuit design, in which a series of one-bit full adders are connected in sequence and the higher output depends on lower carry signals. The delay of RCA is • (•) and the critical path starts from the lowest bit to the last one. However, the probability to activate this critical path is very small [6], [7], which provides the foundation to design speculative-based adder in following researches. In order to obtain the carry signal in advance, for CLA, the real value of carry signals for higher computation block is calculated using signal generation method. The critical path can be reduced efficiently. However, the process of carry signal generation is complicated in CLA, which could produce large logic area and will result in a big power consumption. Based on the idea of CLA, ETAII in [4] makes a full use of paralleling calculation and introduced approximation to reduce the power overhead as shown in Fig.2. The adder is divided into several stages. Each of the stages has a carry generator and a sum generator. The output of one stage comes from its sum generator with the previous carry signal. Thus, the critical path is composed of one sum stage and carry signal generator. However, the carry signal generator involves only parts of the lower input data, which could cause large error when a wrong prediction happens in the upper stage of the adder. For 32-bits ETAII with 4 bits for each stage, the maximum error magnitude could be 228, which is too large that the adder have little practical value in real application.

In [5], an approximate adder with carry skip technique (ACSA) is proposed based on ETAII, in which a multiplexor is used to choose the carry signal for the sum generator in each stage. Different from ETAII, this adder detects the property of carry propagation in previous stage. When all the bits in previous (• − 1) • *h*-stage is

Design And Implementation of a Hybrid LUT/Multiplexer Architectures for FPGA



in carry propagation mode, it will select the carry signal from (• − 2) • *h*-stage. In this way, the error rate of the adder will be decreased. Furthermore, a method for error compensation is also used. However, the error magnitude of this adder is still large. Take 32-bits adder with 4 bits for each stage, the maximum error magnitude could be 220. Meanwhile, the extra multiplexors could consume more area, energy and delay as well.

III. PROJECT DESCRIPTION

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.

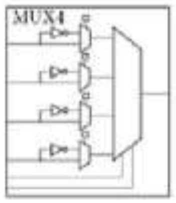


Fig 2: MUX4 LE depicting optional data input inversions

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.

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. 1 is used together with nonfracturable 6-LUTs. This element shares the same number of inputs as

Design And Implementation of a Hybrid LUT/Multiplexer Architectures for FPGA

a 6-LUT lending for fair comparison with respect to the input connectivity.

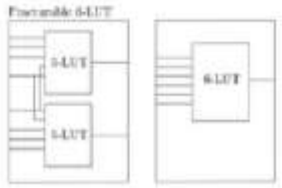


Fig 3: Fracturable 6-LUT that can be fractured into two 5-LUTs with two shared inputs

For the fracturable architecture, we consider an eight-input LE, closely matched with the adaptive logic module in recent Altera Stratix FPGA families. A 6-LUT that can be fractured into two 5-LUTs using eight inputs is shown in Fig. 2. Two five-input functions can be mapped into this LE if two inputs are shared between the two functions. If no inputs are shared, two four-input functions can be mapped to each 5-LUT. For the MUX4 variant, Dual MUX4, we use two MUX4s within a single eight-input LE. In the configuration, shown in Fig. 3, 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.

An architecture in which a 4-to-1 MUX (MUX4) is fractured into two smaller 2-to-1 MUXs was first considered. However, since a 2-to-1 MUX's mapping flexibility is quite limited (can only map two-input functions and the three-input 2-to-1 MUX itself), little benefit was added compared with the overheads of making the MUX4 fracturable and poor area results were observed.

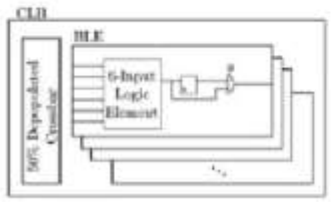
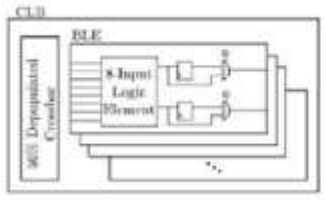


Fig 4: Hybrid CLB with a 50% depopulated intra- CLB crossbar depicting BLE internals for a nonfracturable architecture

IV. 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 [4]. Fig. 4 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. 4 shows the organization of our CLB and internal BLEs.



Design And Implementation of a Hybrid LUT/Multiplexer Architectures for FPGA

Fig 5: Hybrid CLB with a 50% depopulated intra CLB crossbar depicting BLE internals for a fracturable architecture

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 [18]. The same sweep of MUX4 to LUT ratios was also performed. Fig. 5 shows the fracturable architecture with eight inputs to each BLE that contains two optional registers. We evaluate fracturability of LEs versus nonfracturable LEs in the context of MUX4 elements since fracturable LUTs are common in commercial architectures. For example, Altera Adaptive 6-LUTs in Stratix IV and Xilinx Virtex 5 6-LUTs can be fractured into two smaller LUTs with some limitations on inputs.

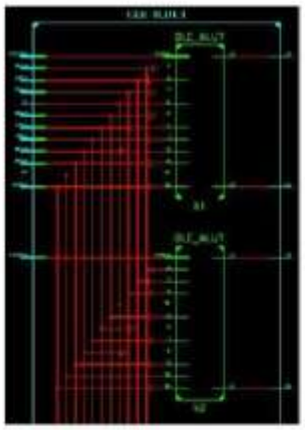
The crossbar for fracturable architectures are larger than the nonfracturable architectures for two reasons. Due to the virtual increase of LEs, a larger number of CLB inputs are required, which increases crossbar size. Since there are now twice as many outputs from the LEs, these additional outputs need to also be fed back into the crossbar, also increasing its size. Due to this disparity in crossbar size, fair comparisons cannot be made between fracturable and nonfracturable architectures. Therefore, in this paper, we compare nonfracturable hybrid CLB architectures to a baseline LUT only nonfracturable architecture and we compare fracturable hybrid CLB architectures to a baseline LUT-only fracturable architecture. Sparse crossbars have been previously studied [19] and in this paper, we model a 50% depopulated crossbar within the CLB for intracluster routing for both nonfracturable and fracturable architectures as compared with the preliminary publication [15] that only modeled a full input crossbar

V.SIMULATION RESULTS

AREA REPORT:-

[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]

RTL SCHEMATIC:-



DELAY ANALYSIS:-

Minimum period: No path found
Minimum input arrival time before clock: 7.560ns
Maximum output required time after clock: 6.631ns
Maximum combinational path delay: 13.418ns

SIMULATION RESULTS:-