A Case for Hardened Multiplexers in FPGAs

S. Alexander Chin and Jason H. Anderson
Department of Electrical and Computer Engineering
University of Toronto
Toronto, Canada
Email: {xan, janders}@eecg.toronto.edu

***Abstract*—This paper presents a case for a hybrid configurable logic block that contains a mixture of LUTs and hardened multiplexers towards the goal of higher logic density and area reduction. Technology mapping optimizations, called *MuxMap*, that target the proposed architecture are implemented using a modified version of the mapper in the ABC logic synthesis tool. VPR is used to model the new hybrid configurable logic block and verify post place and route implementation. Multiple hybrid configurable logic block architectures with varying *MUX:LUT* ratios are evaluated across three benchmark suites with both Quartus II and Odin-II front-end RTL synthesis tools. Experimentally, we show that without any mapper optimizations we naturally save $\sim$4% area post place and route and with *MuxMap* optimizations in ABC yielding $\sim$6% area reduction post place and route while maintaining mapping depth, overall configurable logic block count, and routing demand.**

I. Introduction

Throughout the history of FPGAs, Look-up-Tables (LUTs) have been the primary logic element used to realize combinatorial 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. 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 [1], [2]. Recently, a number of other works have explored alternative FPGA logic element architectures for performance improvement [3], [4], [5], [6], [7] to close the large gap between FPGAs and ASICs [8]. In this paper, we propose incorporating (some) hardened multiplexers in FPGA logic blocks as a means of increasing silicon area efficiency and logic density.

Multiplexer-based logic blocks for 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 [9] in the early '90s. 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 CAD flow. Nevertheless, it is widely understood that LUTs are inefficient at implementing multiplexers, and that multiplexers are frequently used in logic circuits. To underscore the inefficiency of LUTs implementing multiplexers, consider that a 6-LUT is essentially a 64-to-1 multiplexer (to select 1 of 64 truth-table rows) and SRAM configuration cells, yet it can only realize a 4-to-1 multiplexer (4 data + 2 select = 6 inputs).

In this work, we present a 6-input logic element based on a 4-to-1 multiplexer, *MUX4*, that can realize a subset of 6-input Boolean logic functions, and a new hybrid complex logic block (CLB) that contains a mixture of *MUX4*s and 6-LUTs. The proposed *MUX4*s are small compared to a 6-LUT ($\sim$10% of 6-LUT area), and can efficiently map all $\{2, 3\}$-input functions and some $\{4, 5, 6\}$-input functions. We present a CAD flow for mapping into the proposed hybrid CLBs, created using ABC [10] and VPR [11], and describe technology mapping techniques that encourage the selection of logic functions that can be embedded into the *MUX4* elements. In an architecture study, we consider the fraction of logic elements that should be LUTs versus *MUX4*s to optimize logic density.

The main contributions of this work are as follow:

- A hybrid CLB that contains a mixture of *MUX4* logic elements and traditional LUTs that yields an average 6% area savings with a projected 10% for Quartus II synthesized circuits.
- Mapping techniques targeted towards the hybrid CLB architecture that optimizes for area, while preserving original mapping depth.
- A full architecture evaluation with MCNC [12], VTR7 [13], and CHStone [14] benchmarks facilitated by LegUp [15], VTR [13], and Quartus II assisted by the *vqm2blif* translation tool [16].

The remainder of the paper is organized as follows: Section II outlines related work. Section III discusses the proposed *MUX4* logic element. Section IV presents technology mapping approaches to target the proposed hybrid architecture. Section V shows how we modelled the hybrid complex logic blocks in VPR. Section VI discusses our evaluation methodology and gives results. Finally, we close with some final remarks in Section VII.

II. Related Work

Recent works have shown that heterogeneous architectures and synthesis methods can have significant impact on improving logic density and delay, narrowing the ASIC-FPGA gap. Works by Anderson and Wang with "gated" LUTs [4], then with asymmetric LUT logic elements [5], show that LUT elements present in commercial FPGAs provide unnecessary flexibility.

Towards improved delay and area, macro cell-based FPGA architectures have been proposed [6], [7]. These works describe significant changes to traditional FPGA architectures whereas the changes proposed here build on architectures used in industry and academia [1]. Similarly, And-Inverter Cones

978-1-4799-2198-0/13/$31.00 ©2013 IEEE

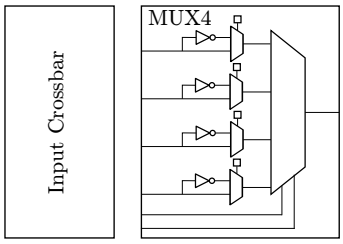

Fig. 1. *MUX4* logic element.

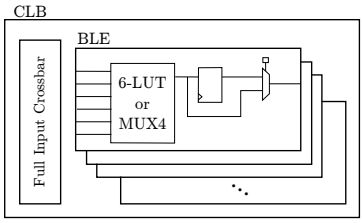

Fig. 2. Hybrid CLB and BLE Internals.

have been proposed as replacements for LUTs, inspired by And-Inverter graphs [3].

Recent work by Purnaprajna and Ienne [17] explored the possibility of re-purposing existing multiplexers contained within Xilinx Logic Slices. 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 VTR benchmarks.

III. Proposed Architecture

A. MUX4: 4-to-1 Multiplexer Logic Element

The *MUX4* logic element shown in Figure 1 consists of a 4-to-1 multiplexer with optional inversion on its inputs to allow the realization of any $\{2, 3\}$-input function, some $\{4, 5\}$-input functions, and one 6-input function – a 4-to-1 multiplexer itself. A 4-to-1 multiplexer matches the input pin count of a 6-LUT, allowing for fair comparisons with respect to connectivity and intra-cluster routing.

Naturally, any two-input Boolean function can be easily implemented in the *MUX4*: the two 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 co-factors 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 co-factors with at most two variables. A second decomposition of the co-factors about one of their two remaining variables produces co-factors with at most one variable. Such single-variable co-factors 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 co-factors with most 1 input-variable.

Observe that input inversion on each select input is omitted. Inversion of the select lines permute the four multiplexer inputs and since our architecture has a full input crossbar, there is no loss of routing flexibility.

B. Hybrid Complex Logic Block

The *MUX4* element is proposed to work in conjunction with 6-LUTs, creating a hybrid CLB with a mixture of non-fracturable 6-LUTs and *MUX4*s, as shown in Figure 2. The hybrid CLB clusters contain ten six-input BLEs that contain either a *MUX4* element or a 6-LUT connected to an optional register. We consider a variety of different architectures, where we vary the ratio of *MUX4*s to LUTs within the ten element CLB from 2:8 to 4:6 *MUX4*s:6-LUTs. Figure 2 illustrates the organization of our CLB and internal BLEs. There are a number of differences between our model and commercial architectures. 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. We currently do not explore the interaction of *MUX4*s with fracturable LUTs, as we keep input routing demand constant between both of our logic elements. In any case, the proposed *MUX4* element is more area efficient at implementing small 3-input functions than a smaller fractured LUT, and for any function with more than 3 inputs, the *MUX4* element is a clear win (if it is able to implement it). We model a full input crossbar within the CLB for intra-cluster routing. Output equivalence of logic elements was modelled in VPR so that each equivalent logic element (*MUX4*s with *MUX4*s, LUTs with LUTs) in the CLB can be swapped. Extended discussion on architecture modelling follows in Section V.

C. FPGA Area Model

A 4-to-1 multiplexer can be realized with three 2-to-1 multiplexers. Hence, the *MUX4* element contains seven 2-to-1 multiplexers and four SRAM cells in total (see Figure 1). The optional inversion uses the four SRAM cells, whereas the rest of the configuration is performed by routing. Additionally, the depth of the multiplexer tree is halved compared to the 6-LUT, which has six 2-to-1 multiplexers on its longest paths. Conservatively, assuming constant pass transistor sizing and that the area of a 2-to-1 multiplexer and 6 transistor SRAM cell are roughly equivalent, the *MUX4* element has $\frac{1}{16}th$ the SRAM area and $\frac{1}{8}th$ the multiplexer area of a 6-LUT. Therefore the area of the *MUX4* is approximately 10% the area of a 6-LUT overall.

Again, this estimate is conservative since we do not account for the larger transistor widths that are necessary for the deeper multiplexer tree within the 6-LUT. An area/delay trade-off can also be seen here: transistors can be sized up in the *MUX4* element reaping delay reduction or as we examine in this work,

reap area reductions while conservatively estimating the delay to be that of a 6-LUT. Although we can estimate the area relative to a 6-LUT, it is important that we estimate global FPGA area and also consider the number of CLB tiles along with the supporting logic and routing area per tile.

Throughout this work, global FPGA area was estimated assuming that, per tile, 50% of the area is inter- and intra-cluster routing, 30% of the area is used for LUTs, and 20% for registers and other miscellaneous logic, following Anderson and Wang [4]. Using this model we can make some observations about the hybrid CLB architecture. The 30% that normally would account for ten 6-LUT logic elements within the tile is now split between the smaller *MUX4* elements and 6-LUTs. For example, in a 3 *MUX4* : 7 6-LUT architecture, the area relative to the reference model can be estimated by deducing the $LogicChange\% = (3 \times 0.10 + 7)/10$, and multiplying $LogicChange\% \times 30\% = 21.9\%$. If routing and miscellaneous area were held constant, our overall architecture area is $Area_{3:7} = 50\% + 20\% + 21.9\% = 91.9\%$ of the reference area – $\sim$8% area savings. However, this is the maximum area savings and it can only be realized by circuits that have a *natural* (i.e. inherent) *MUX4*:LUT ratio greater than or equal to the architecture ratio. And since any function that can be mapped to a *MUX4* element can also be mapped into a 6-LUT, all excess *MUX4* functions can be mapped to 6-LUTs. If the *natural MUX4*:LUT ratio of the circuit is less than the architecture ratio, additional CLBs will be required to supply more LUTs. Additionally, the number of CLBs may also increase during CLB packing ($CLBChange\%$) and routing demand may increase post placement and routing ($RoutingChange\%$). In general, the model used to estimate area relative to the baseline 6-LUT only architecture is as follows:

$$\begin{aligned} Area\% = & CLBChange\% \times \\ & (50\% \times RoutingChange\% + \\ & 30\% \times LogicChange\% + 20\%) \end{aligned} \quad (1)$$

Using this model, it is useful to calculate how many additional CLBs can be tolerated for our new architectures. Again, consider a 3:7 *MUX4*:LUT architecture. Disregarding packing, placement, and routing effects:

$$\begin{aligned} NumCLB_{3:7} \leq & 1/Area_{3:7} \times NumCLB_{LUT} \\ \leq & 1.09 \times NumCLB_{LUT} \end{aligned} \quad (2)$$

This means that an area win can only be achieved if the number of CLBs needed to implement circuits in the hybrid 3:7 architecture is less than 1.09× the number needed for a traditional LUT-only architecture.

IV. Technology Mapping

ABC [10] was used for technology mapping, with modifications that allow for *MUX4*-embeddable function identification and custom mapping. The internal data structure used within ABC is an AND-inverter graph (AIG), where the logic circuit is represented using 2-input AND gates with inverters. *Priority Cuts* mapping in ABC (invoked with the *if* command) [18] was modified to perform our custom technology mapping. This mapper traverses the AIG from primary inputs to primary outputs finding intermediate mappings for internal nodes and finally the primary outputs, using a dynamic programming approach. The priority cuts mapper performs multiple passes on the AIG to find the best cut per node. For depth-oriented mapping, the mapper first prioritizes mapping depth then optimizes for area discarding cuts whose selection would increase the overall depth of the mapped network.

Based on this standard mapper, two mapper variants were produced and evaluated. The first variant, *NaturalMux*, evaluates and identifies internal functions that are *MUX4*-embeddable, agnostic of the target architecture; i.e. this flow uses the default priority cuts mapping and performs a post-processing step to identify *MUX4*-embeddable functions. From this mapping, we can evaluate what area savings are possible without any core mapper changes. The second variant *MuxMap*, area-weights the *MUX4*-embeddable cuts relative to 6-LUT cuts, thereby establishing a preference for selection/creation of *MUX4*-embeddable solutions.

In all mapper variants, cuts that are *MUX4*-embeddable need to be identified, meaning that we must determine whether the logic function implied by such cuts can be implemented in a *MUX4* logic element. The identification function essentially performs a 2-level Shannon decomposition for all combinations of select inputs – a maximum of C_2^6 times for a cut of size 6. For a logic function f, let $Inputs(f)$ represent its variable set, $\{x_0, ..., x_i\}$, and let $f_{x_i x_j}$ represent the Shannon co-factor of f with respect to its variables x_i and x_j. Logic function f can be implemented in a *MUX4* if and only if:

$$|Inputs(f)| \leq 3 \quad (3)$$

or

$$\begin{aligned} & \exists x_i, x_j \in Inputs(f) \text{ such that} \\ & \quad |Inputs(f_{x_i x_j})| \leq 1, \\ & \quad |Inputs(f_{x_i \overline{x_j}})| \leq 1, \\ & \quad |Inputs(f_{\overline{x_i} x_j})| \leq 1, \\ & \quad |Inputs(f_{\overline{x_i x_j}})| \leq 1 \end{aligned} \quad (4)$$

That is, any function with up to three inputs can be implemented in a *MUX4*, and for functions with four or more inputs, there must exist two variables such that the Shannon co-factors with respect to such variables have one or fewer inputs. Note that 1-level Shannon decomposition technique has previously been leveraged for mapping into asymmetric-LUT architectures [5].

A. NaturalMux

NaturalMux mapping invokes the standard priority cuts mapper. Following mapping, we use the approach above to determine if the LUT logic functions in the mapping are *MUX4*-embeddable. For LUTs that are, we "tag" them in the mapped network written out by ABC. This is needed so we can identify which LUTs are *MUX4*-embeddable in the subsequent packing stage.

B. MuxMap

In default ABC technology mapping, each LUT has a unit area of 1.0. In our *MuxMap* approach, we use a lower weight

for the cases where logic functions are *MUX4*-embeddable. Following the area model where 50% of an FPGA tile area is routing, 30% is 6-LUTs and 20% is miscellaneous circuitry (FFs + other), we can derive the weight of a *MUX4* element versus a 6-LUT. Dividing an FPGA tile into ten sub-tiles that contain a single 6-LUT plus the 6-LUT's associated routing and miscellaneous circuitry, the 6-LUT portion of a sub-tile is 3% and the miscellaneous circuitry is 7% of a complete tile. Recall from Section III-C that a *MUX4* element consumes $\sim$10% of the area of a 6-LUT. Therefore, the area of a sub-tile with a *MUX4* is 7.3% of the entire tile. A sub-tile with a *MUX4* would be roughly $7.3\%/10\% = 0.73$ of the area of a sub-tile with a 6-LUT (assuming the routing and other circuity is held constant). Following this reasoning, we weight *MUX4*s conservatively at 0.8 and 6-LUTs at 1.0 during technology mapping.

C. Select Mapping

Depending on the circuit, *NaturalMux* or *MuxMap* may be preferred. In *Select* Mapping, the circuit is first mapped using *NaturalMux*. Following from the discussion in Section III-C, we know that if a circuit's *MUX4*:LUT ratio is higher than the architectural ratio, maximum area reductions are realized. So, if the *Natural* ratio of the circuit is higher than our architectural ratio, we use this mapping. Otherwise, if the *Natural* ratio is lower than the architectural ratio, we re-run mapping with the *MuxMap* mapper to encourage the selection of more *MUX4*-embeddable logic elements.

V. Modelling using VPR

A pre-release version of VPR7 was used to perform architectural evaluation. The standard ten 6-LUT CLB architecture in 40nm included with the VPR distribution was used for the baseline modelling [13]. The hybrid CLB shown in Figure 2 was modelled using the XML-based VPR architectural language. The snippet from the architecture file for the physical block hardened *MUX4* element is shown in Figure 3 – this code specifies a *MUX4* as a 6-input 1-output blackbox to VPR. Additionally, since all *MUX4*s can also be mapped to 6-LUTs, an additional *mode* was added to the 6-LUT physical block, shown in Figure 4. The *mode* concept allows the VPR packer to pack LUTs into LUTs (as usual), but also enables *MUX4*s to be packed into LUTs. Architectures with CLBs having *MUX4*:LUT ratios from 2:8 to 4:6 were created from the baseline 40nm architectures with delays through *MUX4*s pessimistically set equivalent to the 6-LUT delays.

```
<pb_type name='mux4hard' blif_model='.subckt mux4' num_pb='1'>
 <input name='in' num_pins='6'/>
 <output name='out' num_pins='1'/>
</pb_type>
```

Fig. 3. *MUX4* logic element model.

Importantly, we made a small modification to the VPR packing algorithm itself so that it prefers to pack *MUX4* netlist elements into *MUX4* logic elements in the architecture. Without this, the *MUX4* netlist elements might needlessly consume LUTs, which should be reserved, where possible, for those netlist elements that demand their flexibility. Without this modification, a significant CLB usage increase was observed across our benchmark sets.

```
<!— Within LUT Element definition —>
<mode name='6lut'>
 <pb_type name='lut6' blif_model='.names' num_pb='1' class='lut'>
  <input name='in' num_pins='6' port_class='lut_in'/>
  <output name='out' num_pins='1' port_class='lut_out'/>
 </pb_type>
<mode name='mux4'>
 <pb_type name='mux4lut' blif_model='.subckt mux4' num_pb='1'>
  <input name='in' num_pins='6'/>
  <output name='out' num_pins='1'/>
 </pb_type>
```

Fig. 4. *MUX4* mode in the 6-LUT element model.

VI. Experimental Evaluation

Three benchmarks suites were used to evaluate our architectures: MCNC [12], VTR7 [13], and CHStone [14]. Two sets of experiments were performed using two different tool flows over these three benchmark sets. The first set of experiments uses BLIF netlists of the MCNC and VTR7 benchmarks that are pre-synthesized with ODIN-II and distributed with the pre-release of VTR7 [13]. The second set of experiments consist of the CHStone and the Verilog versions of the VTR7 benchmarks. LegUp 3.0 [15] was used for C-to-verilog HLS on the CHStone benchmarks and both benchmarks sets use Altera's Quartus II CAD tool for front-end synthesis from Verilog. The main difference between the two flows is the front-end synthesis tool. ODIN-II-synthesized benchmarks can be put through the entire CAD flow but Quartus II-synthesized circuits contain many blackboxes (e.g. for block RAMs, DSP blocks, etc.) that are not fully supported by VPR architectures. Consequently, for the first set of experiments, we can execute a complete place and route for area assessment; however, for the second set of experiments, we can only project the final area results based on the post-mapping implementation.

A. MCNC & VTR7 Benchmarks

1) Mapping: Using ABC, we performed technology-independent optimization using the standard *resyn2* script included with the ABC distribution [10]. Then, we performed technology mapping with the priority cuts mapper (*if* command), targeting a LUT size of 6. After mapping, we analyzed the LUTs in the mapped list to determine which LUTs are embeddable into *MUX4* logic elements. We then computed the ratio of the number of *MUX4*-embeddable logic elements to the total number of logic elements – i.e. the *Natural MUX4*:LUT ratio. Based on this ratio for each circuit, we projected the area benefits of hybrid architectures 2:8 to 4:6 *MUX4*:LUT ratios. The area projections were made using complete (full) CLB packing, and the tile area breakdown described in Section III-C (assuming no routing area change). The results are shown in Table I.

For architecture ratios of 2:8, 3:7, and 4:6, the lower-bound areas relative to a traditional 6-LUT-based architecture are 94.6%, 91.9%, and 89.2%, respectively. This lower bound can only be achieved when the *Natural MUX4*-embeddable ratio exceeds the architecture ratio. When the *Natural* ratio is lower than the architecture ratio, more CLBs are required, increasing area. Looking at the first half of Table I, the baseline number of logic elements (LEs) along with the *Natural MUX4* ratio is given along with projected areas for each architecture for *NaturalMux*. For the MCNC suite, a large majority of the

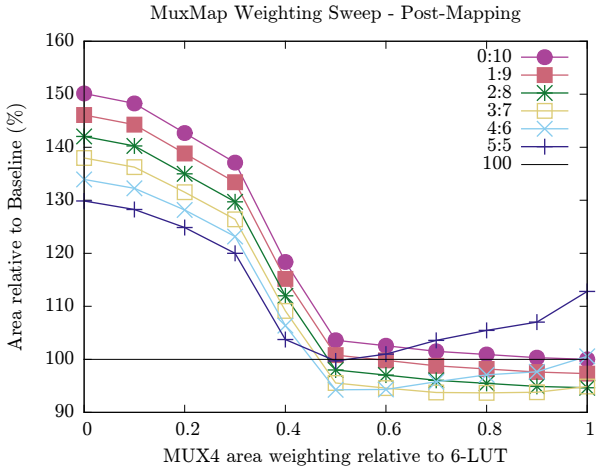

Fig. 5. *MuxMap* varying the weight from zero area to equal 6-LUT area for different *MUX4*:LUT architectures. A weight equal to LUT area is equivalent to the *NaturalMux* mapping. As the *MUX4* area weighting is increased for some architecture ratios (e.g. 5:5 and 4:6), an increase in relative area is seen due unbalanced circuit and architecture *MUX4*:LUT ratios.

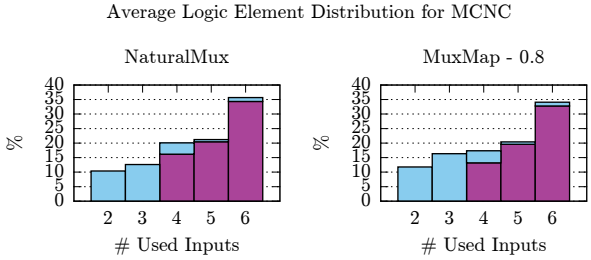

Fig. 6. Used input distribution of MCNC benchmarks with *MUX4*-embeddable logic elements in light-blue and LUTs in dark-purple.

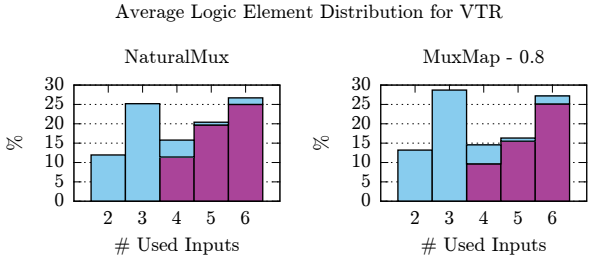

Fig. 7. Used input distribution of VTR benchmarks with *MUX4*-embeddable logic elements in light-blue and LUTs in dark-purple.

benchmarks see full area reduction in a 2:8 architecture – i.e. most have a *MUX4* ratio greater than 0.2. As we increase architecture ratio to 3:7, we see there are some benchmarks that achieve the full area reduction, but others require more 6-LUTs, increasing CLB count. For example, *clma* and *des* see full gains for the 3:7 architecture, while *alu4* shows no area reduction in a 3:7 architecture. Overall, 2:8 is the best architecture with ∼5% projected post-mapping area reduction. VTR7 benchmarks perform better overall compared to MCNC, with a higher *Natural MUX4*-embeddable ratio overall. For the VTR7 benchmarks, a 3:7 architecture works best, yielding ∼7% projected area reduction post-mapping. Combining both suites, either a 2:8 or 3:7 architecture provide a ∼5% reduction.

Our second mapper variant, *MuxMap*, seeks to improve mapping by increasing the number of *MUX4*-embeddable logic functions by reducing the area cost of a *MUX4*-embeddable cut. This increases the *MUX4*-embeddable ratio of each circuit. However, improvements in the ratio may come at the cost of a higher number of logic elements overall. Figure 5 shows a sweep of the weighting of *MUX4*-embeddable logic elements from zero cost to the cost of a 6-LUT over the combined MCNC and VTR7 suites; projected area is shown on the vertical axis (normalized to a traditional LUT-based architecture). Following from Section III-C, area is (not surprisingly) minimized around a weighting of 0.6 to 0.8 for architectures from 2:8 to 4:6. For the rest of the study, a weighting of 0.8 was chosen.

It is interesting to see how the weighting affects the used input-size distribution for LUTs in the circuits. The input-size distributions for the *NaturalMux* and *MuxMap - 0.8* mappings are shown in Figures 6 and 7 for MCNC and VTR benchmarks, respectively. Each bar in the distribution shows the portion of logic elements (with a given number of used inputs) that are *MUX4*-embeddable (in light-blue). The MCNC suite has only a small percentage of functions with more than 3 inputs that are *Naturally MUX4*-embeddable. With *MuxMap - 0.8*, the percentage of large five and six-input functions is marginally reduced with these functions being implemented as three or four *MUX4*-embeddable functions. Looking at the 6-input functions specifically, we observe that in MCNC, very few are *MUX4*-embeddable; whereas, a greater proportion of 6-input functions are embeddable in the VTR7 benchmarks. Recall that the *only* 6-input logic function that is *MUX4*-embeddable is a 4-to-1 multiplexer. Concerning small functions, note that over 35% of functions in the VTR circuits have three or fewer inputs (Figure 7), which bodes well for the proposed hybrid architectures. Additionally, *MuxMap - 0.8* encourages a decrease in five-input elements and an increase in *MUX4*-embeddable three-input elements. Figure 8 shows the overall final distributions of the combined benchmark suites.

The right-hand side of Table I shows projected area results for *MuxMap* mapping with a weighting of 0.8 for architectures with ratios ranging from 2:8 to 4:6. Here we see that for MCNC, we now are able to map to a 3:7 architecture, due to an increase in the number of *MUX4*-embeddable cuts, ∼2%

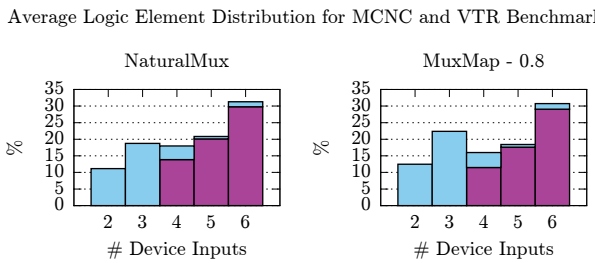

Fig. 8. Used input distribution of the combined VTR & MCNC benchmarks with *MUX4*-embeddable logic elements in light-blue and LUTs in dark-purple.

better than we were able to map to a 2:8 architecture with no specialized mapper changes. For the VTR suite, we also see some improvement over the *Natural* mapping – ∼1%. Over both suites we also realize ∼1% decrease in area for a 3:7 architecture. It is interesting to note that a 2:8 architecture now performs worse than with *Natural* mapping since there is an overall growth of logic elements.

Finally, the *Select* geomean shown in the last row is the overall area reduction if we follow the *Select* mapper strategy outlined in Section IV-C, yielding 7% overall for the 3:7 architecture.

2) Packing, Place & Route: The above results do not consider the possible impact to packing, placement and routability. Therefore, the mapped netlists were packed, placed, and routed using VPR into the architectures discussed in Section V. The final area estimate in Table II is the area reduction relative to the baseline mapper and a traditional 6-LUT architecture. We use the area equation shown in Section III for these estimates. VPR was configured to find minimum channel width with timing driven routing. For most benchmarks, five different placement seeds were used, except for *bgm*, *LU32PEEng*, *LU8PEEng*, *mcml*, and *stereovision2*, due to their 48hr+ runtime. For $CLBChange\%$, we use the change in packed CLBs over the baseline. For $RoutingChange\%$, we use the change in routing area per logic tile (measured in min-width transistors) reported by VPR. Finally, we compute $LogicChange\%$ according to the chosen hybrid CLB architecture ratio – this is constant for a given architecture (e.g. 73% for the 7:3 architecture). The results for each benchmark circuit are shown in Table II.

Overall, we see increases in CLBs for both benchmark suites, but for most architectures this gain is offset by the smaller hybrid CLB area. Again, we show the results of the *Select* mapping strategy and see that over both benchmark suites, marginal gains are seen with the 2:8 architecture since all mapping strategies produced similar post-mapped results. But for the 3:7 architecture, ∼2% gains are seen over the *NaturalMux* mapping. The 3:7 architecture overall yields the greatest reductions in area (6%) post place and route. Compared to post-mapping projections, there is ∼1% degradation over all three architectures when the more accurate post-routing data is accounted for.

B. Quartus II Synthesis: VTR & CHStone

LegUp 3.0 was used for HLS to generate Verilog for the CHStone benchmark suite [14]. Quartus II was then used for RTL synthesis of the VTR7 and CHStone circuits, targeting a Stratix IV architecture. The Quartus synthesis results were written to a VQM (Verilog Quartus Map) file, then converted to BLIF, where the BLIF contained some Altera-specific blackboxes. The BLIF netlists were then carried through ABC exactly like the first set of experiments, including *resyn2*. However, note that the technology mapper ignores the blackboxes. Post-mapping results with area projections are shown in Table III.

The final percentage area savings for the VTR benchmarks, as shown in the "Geomean" row in Table III, are lower than the ODIN-II synthesized benchmarks. On closer inspection, there are six benchmarks that contain less than 250 logic elements

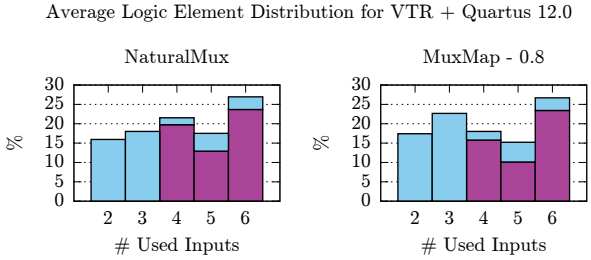

Fig. 9. Final cut distribution for VTR7 benchmarks synthesized with Quartus II with *MUX4*-embeddable cuts in light-blue and LUTs in dark-purple.

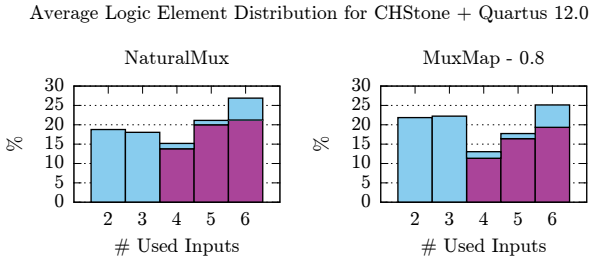

Fig. 10. Final cut distribution for CHStone benchmarks with *MUX4*-embeddable cuts in light-blue and LUTs in dark-purple.

– much smaller than any of the MCNC benchmarks. These benchmarks see this baseline reduction in logic elements from the industrial Quartus II mapper. Again, if we apply our *Select* strategy as well as filter out benchmarks with less than 250 elements, we see comparable *percentage* post-mapping gains to the ODIN-II synthesized circuits.

The CHStone benchmarks perform very well in the proposed architectures. For all circuits, the *Natural MUX4*-embeddable ratio was high enough to realize the gains of the most aggressive 4:6 architecture, with nearly all circuits achieving the physical maximum area savings for each architecture. Figure 10 shows the cut distributions. Here, we can see that there are many 4-to-1 multiplexers within these circuits, likely owing to the sharing of functional units. Although we cannot perform packing/place/route with VPR when Quartus II is used as the front-end, the results above with ODIN-II RTL synthesis demonstrated that most projected post-mapping area reductions are retained post-packing/placement/routing. Assuming this trend holds true with Quartus II synthesis, we project final area reductions of 7% for VTR7 circuits in a 3:7 architecture, and 10% for CHStone circuits in a 4:6 architecture.

VII. Conclusion

We have proposed a new hybrid CLB architecture containing *MUX4* hard multiplexer elements and shown techniques for efficiently mapping to these architectures. Weighting of *MUX4*-embeddable functions with our *MuxMap* technique combined with a *Select* mapping strategy provided aid to circuits with low *Natural MUX4*-embeddable ratios. From our first set of benchmarks synthesized with ODIN-II, area reductions of 6% were seen on average over both MCNC and VTR benchmarks for the 3:7 *MUX4*:LUT architecture.

		Baseline & NaturalMux					MuxMap - 0.8				
		# LEs	*MUX4* Ratio	Arch. M:L Ratio 2:8	3:7	4:6	% LE Change	*MUX4* Ratio	Arch. M:L Ratio 2:8	3:7	4:6
mcnc	alu4	850	0.24	94.6%	100.4%	113.7%	0.6%	0.28	95.2%	95.3%	107.9%
	apex2	1016	0.28	94.6%	94.8%	107.4%	-0.4%	0.33	94.2%	91.5%	98.8%
	apex4	820	0.21	94.6%	104.1%	117.8%	-1.6%	0.28	93.1%	92.5%	104.8%
	bigkey	567	0.20	94.7%	105.1%	119.0%	0.0%	0.20	94.7%	105.1%	119.0%
	clma	3336	0.32	94.6%	91.9%	101.2%	1.4%	0.35	95.9%	93.1%	97.2%
	des	673	0.39	94.6%	91.9%	90.1%	6.4%	0.44	100.6%	97.8%	94.9%
	diffeq	659	0.20	94.6%	104.6%	118.4%	-0.2%	0.34	94.5%	91.8%	97.7%
	dsip	681	0.33	94.6%	91.9%	99.1%	0.0%	0.33	94.6%	91.9%	99.1%
	elliptic	1792	0.38	94.6%	91.9%	92.3%	0.5%	0.48	95.0%	92.3%	89.6%
	ex1010	2738	0.18	96.7%	107.3%	121.5%	0.4%	0.24	95.0%	100.7%	114.0%
	ex5p	638	0.24	94.6%	99.6%	112.8%	0.6%	0.28	95.2%	95.3%	107.9%
	frisc	1764	0.31	94.6%	91.9%	102.0%	0.4%	0.36	95.0%	92.3%	96.2%
	misex3	807	0.27	94.6%	96.1%	108.9%	0.3%	0.32	94.8%	92.1%	101.9%
	pdc	2289	0.23	94.6%	100.5%	113.8%	0.4%	0.29	95.0%	93.4%	105.7%
	s298	667	0.29	94.6%	93.7%	106.1%	0.9%	0.34	95.5%	92.7%	99.0%
	s38417	2229	0.34	94.6%	91.9%	98.0%	1.8%	0.45	96.3%	93.5%	90.8%
	s38584.1	2139	0.39	94.6%	91.9%	90.2%	1.6%	0.48	96.1%	93.4%	90.6%
	seq	941	0.26	94.6%	97.1%	110.0%	1.0%	0.31	95.5%	92.8%	104.0%
	spla	1879	0.24	94.6%	100.3%	113.5%	0.3%	0.27	94.9%	96.1%	108.8%
	tseng	666	0.49	94.6%	91.9%	89.2%	-0.3%	0.51	94.3%	91.6%	88.9%
	Geomean	-	0.28	94.7%	96.8%	105.8%	1.1%	0.33	95.3%	94.2%	100.5%
vtr	bgm	31480	0.25	94.6%	98.9%	112.0%	1.4%	0.29	95.9%	95.2%	107.8%
	blob_merge	6036	0.22	94.6%	102.5%	116.1%	1.0%	0.30	95.6%	92.8%	104.6%
	boundtop	2932	0.43	94.6%	91.9%	89.2%	0.3%	0.50	94.9%	92.2%	89.4%
	ch_intrinsics	382	0.29	94.6%	93.8%	106.2%	0.0%	0.29	94.6%	93.8%	106.2%
	diffeq1	466	0.33	94.6%	91.9%	99.2%	0.9%	0.43	95.4%	92.7%	90.0%
	diffeq2	317	0.32	94.6%	91.9%	100.8%	1.3%	0.34	95.8%	93.1%	99.0%
	LU32PEEng	72905	0.41	94.6%	91.9%	89.2%	0.6%	0.46	95.1%	92.4%	89.7%
	LU8PEEng	21927	0.43	94.6%	91.9%	89.2%	0.6%	0.46	95.2%	92.5%	89.7%
	mcml	99110	0.64	94.6%	91.9%	89.2%	1.9%	0.68	96.4%	93.6%	90.9%
	mkDelayWorker32B	5467	0.43	94.6%	91.9%	89.2%	1.3%	0.47	95.8%	93.1%	90.3%
	mkPktMerge	232	0.74	94.6%	91.9%	89.2%	0.4%	0.76	95.0%	92.3%	89.6%
	mkSMAdapter4B	1997	0.32	94.6%	91.9%	101.2%	1.2%	0.37	95.7%	93.0%	94.4%
	or1200	2955	0.38	94.6%	91.9%	91.7%	2.9%	0.50	97.4%	94.6%	91.8%
	raygentop	1995	0.31	94.6%	91.9%	102.6%	2.9%	0.51	97.4%	94.6%	91.8%
	sha	2215	0.35	94.6%	91.9%	96.7%	0.1%	0.36	94.6%	91.9%	95.0%
	stereovision0	11232	0.76	94.6%	91.9%	89.2%	2.9%	0.86	97.3%	94.5%	91.7%
	stereovision1	10059	0.69	94.6%	91.9%	89.2%	0.2%	0.73	94.8%	92.1%	89.4%
	stereovision2	28596	0.52	94.6%	91.9%	89.2%	1.3%	0.56	95.8%	93.1%	90.4%
	stereovision3	173	0.51	94.6%	91.9%	89.2%	0.6%	0.55	95.1%	92.4%	89.7%
	Geomean	-	0.41	94.6%	92.9%	95.4%	0.7%	0.47	95.7%	93.1%	93.6%
Overall Geomean		-	0.34	94.7%	94.9%	100.6%	0.9%	0.40	95.5%	93.7%	97.1%
Select Geomean		-	-	94.6%	93.0%	96.8%					

TABLE I. POST MAPPING AREA ESTIMATE FOR MCNC & ODIN-II SYNTHESIZED VTR BENCHMARKS ASSUMING COMPLETE CLB PACKING AND NO INCREASE IN ROUTING DEMAND.

		Baseline			NaturalMux						MuxMap - 0.8					
		Natural *MUX4* Ratio	# CLBs	Routing	Arch. 2:8 % CLBs	Arch. 2:8 % Routing	Arch. 3:7 % CLBs	Arch. 3:7 % Routing	Arch. 4:6 % CLBs	Arch. 4:6 % Routing	Arch. 2:8 % CLBs	Arch. 2:8 % Routing	Arch. 3:7 % CLBs	Arch. 3:7 % Routing	Arch. 4:6 % CLBs	Arch. 4:6 % Routing
mcnc	alu4	0.24	85	5386	102.4%	96.4%	110.6%	95.9%	128.2%	91.8%	102.4%	100.0%	110.6%	95.9%	128.2%	91.8%
	apex2	0.28	102	7840	100.0%	90.3%	104.9%	85.6%	120.6%	79.9%	101.0%	90.0%	104.9%	85.6%	120.6%	79.9%
	apex4	0.21	83	7806	103.6%	91.3%	114.5%	87.3%	131.3%	81.7%	98.8%	91.3%	114.5%	87.3%	131.3%	81.7%
	bigkey	0.20	57	4563	100.0%	99.4%	114.0%	100.6%	133.3%	100.6%	100.0%	99.4%	114.0%	100.6%	133.3%	100.6%
	clma	0.32	334	10272	100.0%	95.3%	100.9%	91.4%	115.0%	87.3%	101.5%	91.6%	100.9%	91.4%	115.0%	87.3%
	des	0.39	68	4628	101.5%	101.8%	100.0%	100.6%	102.9%	99.4%	105.9%	104.7%	100.0%	100.6%	102.9%	99.4%
	diffeq	0.20	67	4911	101.5%	109.3%	114.9%	111.6%	132.8%	104.1%	101.5%	100.6%	114.9%	111.6%	132.8%	104.1%
	dsip	0.33	69	4730	100.0%	98.8%	100.0%	98.8%	110.1%	100.0%	100.0%	98.8%	100.0%	98.8%	110.1%	100.0%
	elliptic	0.38	180	7057	100.0%	96.9%	101.1%	97.8%	107.8%	102.3%	100.6%	93.0%	101.1%	97.8%	107.8%	102.3%
	ex1010	0.18	274	11884	105.8%	89.5%	117.5%	87.9%	136.5%	78.8%	103.3%	95.8%	117.5%	87.9%	136.5%	78.8%
	ex5p	0.24	64	7326	101.6%	84.1%	110.9%	86.4%	126.6%	88.2%	101.6%	88.6%	110.9%	86.4%	126.6%	88.2%
	frisc	0.31	178	9269	101.1%	97.7%	102.8%	96.8%	116.3%	94.2%	101.7%	97.9%	102.8%	96.8%	116.3%	94.2%
	misex3	0.27	82	5957	100.0%	96.4%	106.1%	93.7%	120.7%	91.4%	100.0%	101.1%	106.1%	93.7%	120.7%	91.4%
	pdc	0.23	229	10787	103.1%	93.7%	112.2%	85.2%	127.9%	80.9%	101.7%	91.2%	112.2%	85.2%	127.9%	80.9%
	s298	0.29	68	6697	100.0%	92.6%	104.4%	93.6%	119.1%	93.4%	100.0%	96.5%	104.4%	93.6%	119.1%	93.4%
	s38417	0.34	261	4908	100.4%	103.8%	106.9%	107.1%	119.5%	116.9%	102.3%	103.3%	106.9%	107.1%	119.5%	116.9%
	s38584.1	0.39	233	6143	100.4%	100.9%	102.1%	103.2%	109.0%	107.2%	101.7%	99.5%	102.1%	103.2%	109.0%	107.2%
	seq	0.26	95	8009	101.1%	98.9%	106.3%	82.4%	122.1%	85.7%	101.1%	99.3%	106.3%	82.4%	122.1%	85.7%
	spla	0.24	188	9270	102.7%	96.2%	110.6%	86.5%	127.7%	82.0%	101.6%	94.4%	110.6%	86.5%	127.7%	82.0%
	tseng	0.49	67	4767	100.0%	101.8%	100.0%	102.4%	100.0%	105.4%	100.0%	99.4%	100.0%	102.4%	100.0%	105.4%
	Geomean	-	-	-	101.2%	96.6%	106.9%	94.4%	119.9%	93.0%	101.3%	96.7%	104.4%	95.4%	113.7%	95.1%
vtr	bgm	0.25	3599	8229	102.5%	122.2%	112.7%	139.6%	130.9%	138.3%	102.8%	116.1%	109.5%	150.4%	126.9%	144.2%
	blob_merge	0.22	604	7649	102.3%	105.4%	113.6%	102.2%	130.6%	101.2%	101.2%	101.4%	113.6%	102.2%	130.6%	130.6%
	boundtop	0.43	300	5271	100.0%	94.9%	100.0%	104.8%	102.0%	111.8%	100.3%	94.9%	100.0%	104.8%	102.0%	102.0%
	ch_intrinsics	0.29	39	4751	100.0%	101.3%	105.1%	101.9%	120.5%	101.9%	100.0%	101.3%	105.1%	101.9%	120.5%	120.5%
	diffeq1	0.33	47	5819	100.0%	102.3%	100.0%	106.6%	110.6%	100.9%	102.1%	100.0%	100.0%	106.6%	110.6%	110.6%
	diffeq2	0.32	32	5869	100.0%	94.8%	100.0%	94.8%	115.6%	95.3%	103.1%	97.2%	100.0%	94.8%	115.6%	115.6%
	LU32PEEng	0.41	8555	12548	100.7%	117.8%	102.9%	139.5%	-	-	101.0%	108.7%	102.0%	121.8%	-	-
	LU8PEEng	0.43	2579	8833	100.8%	112.4%	102.6%	128.2%	110.6%	139.0%	101.3%	105.8%	102.4%	117.4%	107.4%	139.5%
	mcml	0.64	10422	9467	100.3%	114.5%	100.5%	101.5%	101.2%	112.1%	102.0%	100.2%	102.3%	97.6%	102.7%	97.6%
	mkDelayWorker32B	0.43	550	6411	100.0%	99.6%	100.9%	106.3%	104.2%	113.0%	101.3%	101.7%	100.9%	106.3%	104.2%	104.2%
	mkPktMerge	0.74	24	5357	100.0%	101.9%	100.0%	100.0%	100.0%	100.3%	100.0%	95.7%	100.0%	100.0%	100.0%	100.0%
	mkSMAdapter4B	0.32	204	5738	100.0%	99.5%	102.5%	98.6%	115.7%	98.9%	101.5%	100.0%	102.5%	98.6%	115.7%	115.7%
	or1200	0.38	297	7474	100.0%	96.2%	100.7%	97.9%	107.4%	97.2%	102.7%	93.7%	100.7%	97.9%	107.4%	107.4%
	raygentop	0.31	240	6297	102.5%	100.4%	109.2%	108.7%	124.6%	104.7%	103.3%	101.7%	109.2%	108.7%	124.6%	124.6%
	sha	0.35	226	4532	100.0%	103.0%	101.3%	125.4%	110.2%	138.9%	100.0%	109.0%	101.3%	125.4%	110.2%	110.2%
	stereovision0	0.76	1494	4770	100.8%	102.9%	101.8%	114.3%	103.3%	115.9%	102.5%	98.4%	101.8%	114.3%	103.3%	103.3%
	stereovision1	0.69	1336	7909	100.4%	102.9%	100.7%	103.6%	101.5%	106.1%	100.5%	103.6%	100.7%	103.6%	101.5%	101.5%
	stereovision2	0.52	3584	12589	102.1%	106.4%	105.0%	109.5%	111.4%	119.4%	103.1%	103.1%	105.1%	106.4%	105.3%	101.9%
	stereovision3	0.51	19	3296	105.3%	102.0%	100.0%	101.0%	110.5%	94.3%	105.3%	99.0%	100.0%	101.0%	110.5%	110.5%
	Geomean	-	-	-	109.9%	102.7%	103.1%	115.9%	111.3%	109.6%	101.8%	101.7%	102.9%	105.9%	108.0%	109.5%
Overall Geomean					101.1%	100.5%	105.0%	98.8%	115.8%	99.5%	101.5%	101.0%	103.7%	99.6%	111.0%	98.3%
Area		-			95.9%		95.9%		103.0%		96.6%		95.1%		98.0%	
Select Geomean		-	-	-	101.0%	100.3%	103.5%	97.8%	111.2%	97.8%						
Area		-			95.7%		94.0%		98.0%							

TABLE II. POST PLACE & ROUTE RESULTS WITH FINAL AREA. ALL BENCHMARKS WERE AVERAGED OVER FIVE PLACEMENT SEEDS EXCEPT FOR THAT LARGEST MOST COMPUTE DEMANDING CIRCUITS: BGM, LU32PEENG, LU8PEENG, MCML, STEREOVISION2. LU32PEENG FOR THE 4:6 ARCHITECTURE RAN FOR OVER 100 HOURS WITHOUT COMPLETION.

		Baseline & NaturalMux					MuxMap - 0.8				
		# LEs	*MUX4* Ratio	Arch. M:L Ratio 2:8	3:7	4:6	% LE Change	*MUX4* Ratio	Arch. M:L Ratio 2:8	3:7	4:6
vtr	bgm	12494	0.50	94.6%	91.9%	89.2%	0.78%	0.52	95.3%	92.6%	89.9%
	blob_merge	2363	0.26	94.6%	96.7%	109.5%	-0.13%	0.40	94.5%	91.8%	89.2%
	boundtop	974	0.31	94.6%	91.9%	102.8%	0.42%	0.46	95.0%	92.3%	89.6%
	ch_intrinsics	202	0.70	94.6%	91.9%	89.2%	0.00%	0.71	94.6%	91.9%	89.2%
	diffeq1	36	0.06	111.3%	123.6%	139.9%	0.00%	0.06	111.3%	123.6%	139.9%
	diffeq2	132	0.11	104.7%	116.3%	131.7%	0.00%	0.11	104.7%	116.3%	131.7%
	LU32PEEng	59981	0.48	94.6%	91.9%	89.2%	0.53%	0.54	95.1%	92.4%	89.7%
	LU8PEEng	16075	0.52	94.6%	91.9%	89.2%	0.45%	0.58	95.0%	92.3%	89.6%
	mcml	26150	0.56	94.6%	91.9%	89.2%	0.51%	0.59	95.1%	92.4%	89.7%
	mkDelayWorker32B	827	0.31	94.6%	91.9%	102.8%	0.18%	0.35	94.8%	92.1%	97.0%
	mkPktMerge	236	0.75	94.6%	91.9%	89.2%	0.43%	0.76	95.0%	92.3%	89.6%
	mkSMAdapter4B	215	0.73	94.6%	91.9%	89.2%	0.00%	0.73	94.6%	91.9%	89.2%
	or1200	1919	0.41	94.6%	91.9%	89.2%	1.75%	0.52	96.3%	93.5%	90.8%
	raygentop	787	0.33	94.6%	91.9%	99.0%	0.89%	0.59	95.4%	92.7%	90.0%
	sha	477	0.52	94.6%	91.9%	89.2%	0.21%	0.53	94.8%	92.1%	89.4%
	stereovision0	326	0.65	94.6%	91.9%	89.2%	0.33%	0.66	94.9%	92.2%	89.5%
	stereovision1	443	0.70	94.6%	91.9%	89.2%	0.00%	0.70	94.6%	91.9%	89.2%
	stereovision2	6979	0.26	94.6%	97.5%	110.4%	1.42%	0.31	95.9%	93.2%	104.0%
	stereovision3	69	0.11	105.5%	117.1%	132.7%	3.08%	0.51	97.5%	94.7%	91.9%
	Geomean	-	0.36	96.5%	96.3%	99.3%	0.57%	0.44	96.5%	95.0%	94.9%
Select Geomean				97.0%	96.2%	98.4%					
Select Geomean >250 LEs				94.6%	92.6%	94.3%					
CHStone	adpcm	10379	0.43	94.6%	91.9%	89.2%	2.92%	0.51	97.4%	94.6%	91.8%
	aes	10259	0.43	94.6%	91.9%	89.2%	4.46%	0.56	98.8%	96.0%	93.2%
	blowfish	7812	0.50	94.6%	91.9%	89.2%	0.61%	0.54	95.2%	92.5%	89.7%
	dfadd	4523	0.53	94.6%	91.9%	89.2%	0.53%	0.56	95.1%	92.4%	89.7%
	dfdiv	7376	0.40	94.6%	91.9%	89.4%	4.22%	0.49	98.6%	95.8%	93.0%
	dfmul	2470	0.48	94.6%	91.9%	89.2%	1.82%	0.53	96.3%	93.6%	90.8%
	dfsin	14155	0.46	94.6%	91.9%	89.2%	2.61%	0.53	97.1%	94.3%	91.5%
	gsm	6993	0.44	94.6%	91.9%	89.2%	3.09%	0.51	97.5%	94.7%	92.0%
	jpeg	25110	0.39	94.6%	91.9%	90.0%	5.06%	0.52	99.4%	96.5%	93.7%
	mips	2193	0.48	94.6%	91.9%	89.2%	2.26%	0.54	96.7%	94.0%	91.2%
	motion	3004	0.44	94.6%	91.9%	89.2%	2.04%	0.51	96.5%	93.8%	91.0%
	sha	8342	0.42	94.6%	91.9%	89.2%	4.84%	0.53	99.2%	96.3%	93.5%
	Geomean	-	0.45	94.6%	91.9%	89.3%	2.9%	0.53	97.3%	94.5%	91.8%

TABLE III. POST MAPPING AREA ESTIMATE FOR QUARTUS II SYNTHESIZED CHSTONE & VTR BENCHMARKS ASSUMING COMPLETE CLB PACKING AND NO INCREASE IN ROUTING DEMAND.

Additionally, post place and route degradations of around $\sim$1% were observed from the ideal projected area savings. Our second set of benchmarks synthesized with Quartus II showed that over the VTR benchmark suite, percentage gains were comparable to gains seen in the ODIN-II synthesized benchmarks with 7% projected area savings. CHStone benchmarks high-level synthesized with LegUp show excellent area savings all around, approaching the physical area gains of our hybrid CLB architectures with 10% projected area savings. Overall, this research shows that there is significant potential in exploring mixed multiplexer and LUT architectures for improving logic-density and closing the FPGA-ASIC gap.

Future work includes exploring the interaction of *MUX4*s with fracturable LUTs, as well as developing architectures to fully place and route the Quartus II synthesized circuits. While this study has been primarily area driven, techniques for improving mapping using the reduced delay of *MUX4*s is left to be explored. Lastly, it will be interesting to see how much more these hard multiplexers may facilitate increased functional unit sharing within LegUp.

REFERENCES

[1] E. Ahmed and J. Rose, "The effect of LUT and cluster size on deep-submicron FPGA performance and density," *IEEE Transactions on VLSI*, vol. 12, no. 3, pp. 288–298, 2004.

[2] J. Rose, R. Francis, D. Lewis, and P. Chow, "Architecture of field-programmable gate arrays: the effect of logic block functionality on area efficiency," *IEEE Journal of Solid-State Circuits*, vol. 25, no. 5, pp. 1217–1225, 1990.

[3] H. Parandeh-Afshar, H. Benbihi, D. Novo, and P. Ienne, "Rethinking FPGAs: elude the flexibility excess of LUTs with and-inverter cones," in *ACM/SIGDA FPGA*, 2012, pp. 119–128.

[4] J. Anderson and Q. Wang, "Improving logic density through synthesis-inspired architecture," in *IEEE FPL*, 2009, pp. 105–111.

[5] ——, "Area-efficient FPGA logic elements: architecture and synthesis," in *ASP DAC*, 2011, pp. 369–375.

[6] J. Cong, H. Huang, and X. Yuan, "Technology mapping and architecture evalution for k/m-macrocell-based FPGAs," *ACM Trans. Des. Autom. Electron. Syst.*, vol. 10, no. 1, pp. 3–23, Jan. 2005.

[7] Y. Hu, S. Das, S. Trimberger, and L. He, "Design, synthesis and evaluation of heterogeneous FPGA with mixed LUTs and macro-gates," in *IEEE ICCAD*, 2007, pp. 188–193.

[8] I. Kuon and J. Rose, "Measuring the Gap Between FPGAs and ASICs," *IEEE ICCAD*, vol. 26, no. 2, pp. 203–215, 2007.

[9] K. Karplus, "Amap: A technology mapper for selector-based field-programmable gate arrays," in *DAC*, 1991, pp. 244–247.

[10] A. Mishchenko, S. Chatterjee, and R. Brayton, "DAG-aware AIG Rewriting A Fresh Look at Combinational Logic Synthesis," in *DAC*, 2006, pp. 532–535.

[11] V. Betz and J. Rose, "VPR: A new packing, placement and routing tool for FPGA research," in *FPL*, 1997, pp. 213–222.

[12] S. Yang, "Logic Synthesis and Optimization Benchmarks User Guide Version 3.0," Microelectronics Centre of North Carolina, Research Triangle Park, NC, USA, Jan. 1991.

[13] J. Rose, J. Luu, C. W. Yu, O. Densmore, J. Goeders, A. Somerville, K. B. Kent, P. Jamieson, and J. Anderson, "The VTR project: architecture and CAD for FPGAs from verilog to routing," in *ACM/SIGDA FPGA*, 2012, pp. 77–86.

[14] S. H. Yuko Hara, Hiroyuki Tomiyama and H. Takada, "Proposal and Quantitative Analysis of the CHStone Benchmark Program Suite for Practical C-based High-level Synthesis," *Journal of Information Processing*, vol. 17, pp. 242–254, 2009.

[15] A. Canis, J. Choi, M. Aldham, V. Zhang, A. Kammoona, J. H. Anderson, S. Brown, and T. Czajkowski, "LegUp: High-level synthesis for FPGA-based processor/accelerator systems," in *ACM/SIGDA FPGA*, 2011, pp. 33–36.

[16] K. E. Murray, S. Whitty, S. Liu, J. Luu, and V. Betz, "TITAN: Enabling Large and Complex Benchmarks in Academic CAD," in *FPL*, 2013.

[17] M. Purnaprajna and P. Ienne, "A Case for Heterogeneous Technology-Mapping: Soft Versus Hard Multiplexers," in *IEEE FCCM*, 2013, pp. 53–56.

[18] A. Mishchenko, S. Cho, S. Chatterjee, and R. Brayton, "Combinational and sequential mapping with priority cuts," in *IEEE ICCAD*, 2007, pp. 354–361.