A Custom-Cell Identification Method for High-Performance Mixed Standard/Custom-Cell Designs

Jennifer Y.-L Lo, Wu-An Kuo, Allen C.-H. Wu and TingTing Hwang
Computer Science Department, Tsing Hua University
Hsin-Chu, Taiwan 30043

Abstract:
Over the years, many design methodologies/tools and layout architectures have been developed for datapath-oriented designs. One commonly used approach for high-speed datapath designs is the full-custom design method targeted to bit-sliced or bit-alignment layout architectures. Using this approach, designers can fully exploit design properties, such as various circuit designs, structural regularities and layout structures, to develop high-speed, low-power and high-density datapath designs. However, the main drawbacks of this approach are threefold: (1) extremely high development cost, (2) very long development time, and (3) difficult to migrate the custom design to a new technology.

The other approach is the HDL-based synthesis method targeted to the standard-cell layout architecture. This approach is highly productive and easy to implement. However, due to lack of the ability to exploit the regularity properties of datapath designs, it's mostly suitable to produce low/medium-speed datapath designs. Another design method targeted to mixed standard- and custom-cells is carried out as follows. First, it generates a standard-cell design to obtain a maximal achievable speed. Second, it performs a cell customization process to determine which standard cells need to be customized in order to satisfy the given timing constraint. This cell customization process is usually performed incrementally based on designers' intuitions and experiences. Once designers select a set of standard cells to be customized, they will define the specification of the custom cells and develop the cells accordingly. Finally, the designers will replace the standard cells with the custom ones and re-evaluate the speed of the design. The customization process will continue until the timing constraint is satisfied.

In this paper, we present a custom-cell identification method for high-speed datapath-oriented designs. The proposed method uses an HDL-based standard-cell design flow by integrating a custom-cell identification algorithm to determine a minimal-cost cell set and the design budgets for cell customization. Experimental results on a set of benchmarking designs are reported to demonstrate the effectiveness of the proposed method. Note that our problem is different from the cell-sizing problem. While the cell-sizing problem is to select and to replace (re-size) cells (instances of cells in cell library) one by one, our problem is to select and re-design cells in the cell library and to replace all instances of the cell in the design at one time.

1. A High-Performance Mixed Standard-/Custom-cell Design Flow

The custom-cell identification method for high-performance mixed standard-/custom-cell datapath-oriented designs is defined as: Given an RTL design, a target standard-cell library and a given timing constraint, determine a minimal cell-set for customization and cells' design budgets such that after replacing the standard cells with those custom cells, the final timing of the design satisfies the given timing constraint.

Figure 1 shows the proposed design flow. The inputs to the design flow include an RTL design description in Verilog and a timing constraint. First, we perform the maximum-timing-driven RTL/logic synthesis to convert the RTL design into a hierarchical gate-level design. The main objective of the maximum-timing-driven RTL/logic synthesis is to achieve a design with the maximum speed. In our approach, we use a commercial synthesis tool to synthesize the RTL design into a gate-level design with the maximum speed option. However, when imposing such a tight timing constraint during the synthesis process, the resultant design is usually overly constrained and often results in a design with excessive area costs. To alleviate this problem, we apply an area-driven re-synthesis procedure to minimize the area cost while maintaining the achievable maximum speed.

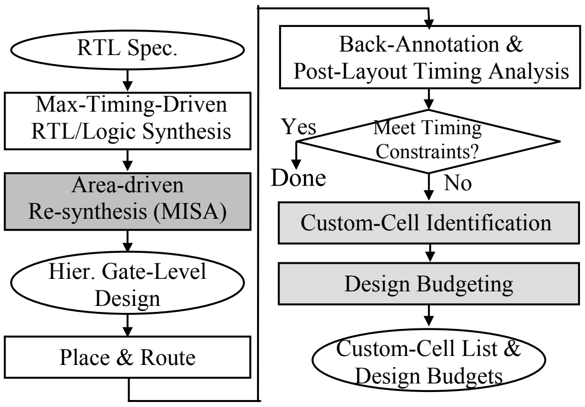
RTL Spec.
Max-Timing-Driven RTL/Logic Synthesis
Area-driven Re-synthesis (MISA)
Hier. Gate-Level Design
Place & Route
Back-Annotation & Post-Layout Timing Analysis
Meet Timing Constraints?
Yes
Done
No
Custom-Cell Identification
Design Budgeting
Custom-Cell List & Design Budgets

Figure 1: The proposed design flow.

The area-driven re-synthesis procedure is described as below. After synthesizing the RTL design into a gate-level netlist, we transform the hierarchical gate-level netlist into a super-graph and extract the timing information for each super-node (i.e., for each module). We then apply the maximal-independent-set based algorithm (MISA) to perform slack assignments and thus determine the delay budget for each module. After determining the delay budget for each module, we uses a commercial tool to re-synthesize the RTL design into a gate-level design by applying the obtained delay budget as the timing constraint for each module.

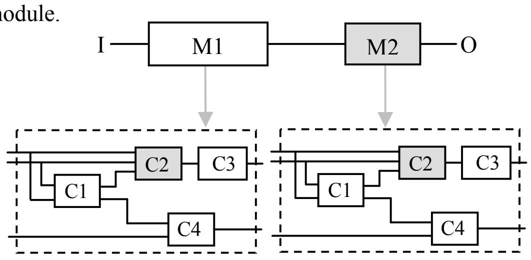

Figure 2: A custom-cell identification example.

Next, we perform place & route tasks and then back-annotate the post-layout timing and RC information. If the design meets the timing constraints, the design process is terminated. Otherwise, we perform the custom-cell identification procedure to determine the cells for customization and their design budgets.

The custom-cell identification procedure is performed as follows. We first locate the paths that violate the given timing constraint. Then, we apply a custom-cell identification

This research was supported by grants of NSC 91-2215-E-007-040 and NSC90-2215-E-007-043.

algorithm to identify the cells for customization such that they will contribute the maximal delay reduction with the minimal customization cost. We can perform the custom-cell procedure at module/cell levels in a hierarchical fashion. For instance, as shown in Figure 2, module *M2* is selected for customization. We can further apply the custom-cell identification algorithm to select cells in the module for customization. As shown in Figure 2, cell *C2* is selected for customization. Note that after we develop a custom-cell of *C2*, we can replace all the standard cells that have the same component type as *C2* with the custom one (e.g., *C2* in *M1*) for timing improvement. Our goal is to determine a minimal-cost cell set for customization. Finally, based on the post-layout timing and RC-extraction information, we can determine the specifications (e.g., driven loads and delay budgets) of the cells for customization. We will discuss the algorithm in details in the next section.

Table 1: The maximum-timing-driven RTL/logic synthesis results.

	Delays (ns)	Max-timing (area/#cell)	Resyn(MISA) (area/#cell)	Area (%)
MAC	3.23	18697/7832	16448/7288	12.03
CPU	2.81	4218/1728	4063/1666	3.67
MCPU	5.46	16860/5620	15875/5379	5.84
EWF	2.71	3666/1068	3912/1171	-6.7
MCU	7.35	18683/7444	17250/7054	7.67
FIR	4.5	31849/12478	28860/11871	9.38
Blowfish	3.08	14391/3863	14233/3837	1.1

2. Experimental Results

We have tested seven benchmarking designs as shown in Table 1. All seven benchmarking designs are RTL designs described in Verilog. In the experiment, we used a 0.35um cell library.

The experiments consist of two parts: (1) the maximum-timing-driven RTL/logic synthesis method and (2) the custom-cell identification algorithm. For the maximum-timing-driven RTL/logic synthesis experiment, we used Synopsys's *Design Compiler* to synthesize the RTL design into a gate-level design with the maximum-speed option. Then, we used AVANTI's *Apollo* to perform the place & route design tasks. After that, we back-annotated the capacitance load information from the layout, follows by using *Design Time* to perform post-layout timing analysis. Subsequently, we invoked the MISA algorithm to determine the delay budget for each module in the RTL design. Finally, we used *Design Compiler* to re-synthesize the RTL design into a gate-level design. In this re-synthesis process, we assigned the pre-determined delay budget as the timing constraint to each module. Table 1 shows the results produced by our proposed maximum-timing-driven RTL/logic synthesis method. The results show that using the MISA-based re-synthesis method we can achieve the maximum timing with an average of 4.7% area reduction.

For the custom-cell identification experiment, we used the maximum achievable timing (Table 1) as the baseline and gradually tightened the timing constraint. Finally, we invoked the custom-cell identification algorithm to determine the minimum custom-cell set and the design budgets. We have compared our heuristic to a branch-and-bound algorithm to demonstrate the effectiveness of our proposed heuristic. Table 2 shows the custom-cell identification results for the MAC design using a heuristic (Heur) and a branch-and-bound algorithm (BB), where #C/A denotes the number of custom cells and the area cost, respectively. The result shows that we need to replace two cells in this module (*cg01* and *xr03* as shown in Table 3) with area cost of 7.33 to satisfy the timing constraint of 2.74ns.

Table 3 shows the delay budgets for the custom modules/cells of the MAC design in order to satisfy the given timing constraint of 2.74ns. For example, the delay from input *in2* to output *carry* of the *fulladder* should be 0.171ns for the custom-cell implementation instead of 0.190ns for the standard-cell implementation. Table 4 shows the cell replacements and load distribution for the MAC design. For example, in order to satisfy the timing constraint of 2.74ns, we need to replace 163 *fulladders* in the MAC design and the driven loads of those *fulladders* are ranged between 0.029 to 0.325pf.

3. Conclusions

In this paper, we have presented a new custom-cell identification method for high-performance datapath-oriented designs targeted to mixed standard-/custom-cell design style. By integrating commercial CAD tools and the proposed custom-cell identification algorithms, we have developed a performance-driven datapath design flow. The experimental results have shown that our proposed method can effectively identify the custom-module/cell-set for customization as well as determine the delay budgets. This provides designers with invaluable information to process their cell customization task that can greatly reduce the risk of design customization.

Table 2: The custom-cell identification results (MAC).

Tconst	BB[#C/A]	Heur[#C/A]
2.98	2/7.33	2/7.33
2.74	2/7.33	2/7.33
2.49	5/12.66	5/12.66
2.24	13/34.32	15/41.66

Table3: Delay budgets for the custom modules/cells of the MAC design with T_{const} =2.74ns.

	Type	Input	Output	Delay budget spec
Custom modules	fulladder	in2	carry	0.190 => 0.171
		in2	sum	0.200 => 0.180
		in3	sum	0.200 => 0.180
		in3	carry	0.190 => 0.171
		in1	carry	0.220 => 0.198
		in1	sum	0.350 => 0.315
Custom cells	cg01	in	out	0.220 => 0.110
	xr03	in	out	0.350 => 0.315

Table 4: The number of cell replacements and load distribution of the MAC design.

T_{const}	Mod	# of replace	Max C_{load}	Min C_{load}	Mean C_{load}
2.98	fulladder	86	0.204	0.029	0.088
2.74	fulladder	163	0.325	0.029	0.085
2.49	fulladder	196	0.325	0.029	0.084
	CLA_10	1	0.186	0.186	0.186
2.24	fulladder	221	0.325	0.024	0.083
	CLA_10	1	0.186	0.186	0.186