Library-free Structure Recognition for Analog Circuits

Maximilian Neuner, Inga Abel and Helmut Graeb
Chair of Electronic Design Automation, Technical University of Munich, Munich, Germany
maximilian.neuner@tum.de, inga.abel@tum.de, helmut.graeb@tum.de

***Abstract*—Extracting structural information of a design is one crucial aspect of many circuit verification and synthesis methods. State-of-the-art structure recognition methods use a predefined building block library to identify the basic building blocks in a circuit. However, the capability of these algorithms is limited by the scope, correctness and completeness of the provided library. This paper presents a new method to automatically generate the recognition rules required to identify a given circuit topology in a large design. Device pairs are grouped into building blocks by analyzing their characteristics, e.g., their connectivity, to enable a structure recognition as unambiguous as possible. The resulting blocks are consecutively assembled to larger blocks until the full building block description of the given topology has been established. Building block libraries dedicated to one specific topology type, e.g., operational amplifiers, can be obtained by applying the method to its basic version, subsequently extending the generated library by the additional elements required to identify its topology variants using the presented method. Experimental results for six folded cascode amplifier and five level shifter topologies are given.**

***Index Terms*—building blocks, structure recognition, library-free, circuit verification**

I. Introduction

Analog circuit verification and synthesis tools usually incorporate expert knowledge for efficient operation. Circuit optimization tools, e.g., Muneda WiCkeD [1] or Cadence Virtuoso ADE XL [2], for example require additional sizing constraints to ensure the proper functionality of the circuit's building blocks, e.g., differential pairs and current mirrors, to compute technical meaningful solutions [3].

Structure recognition methods, e.g., [3]–[5], automatically identify the analog basic building blocks and generate their corresponding sizing rules. These approaches use a hierarchically organized building block library which provides the recognition rules of each included library element of its corresponding level. The lowest level contains building blocks formed by two transistors, higher level building blocks are formed by two sub-blocks from lower hierarchy levels. Structure recognition is also required to set up symmetry constraints for analog layout [6], [7].

The capabilities of the structure recognition method are mainly limited by the completeness of the provided building block library and the correctness of the specified recognition rules. The library provided by [4] only contains well-known analog building blocks consisting of up to six devices. This prevents the identification of larger structures or entire circuit topologies, e.g., operational amplifiers or level shifters, in a complete design. However, this information can be important for circuit verification and synthesis. For example, interface circuits between voltage domains are susceptible to electrostatic discharge (ESD) events [8]. Hence, identifying such interface circuits in a large design is an important aspect of ESD circuit verification.

To overcome this problem, the used building block library could be extended by manually adding new hierarchy levels and building blocks to it, until one specific topology, e.g., a conventional voltage level shifter, can be identified. In practice, this approach is not feasible as a large variety of different circuit topologies exists and the manual implementation of the corresponding recognition rules is error prone and tedious. Instead, automatic approaches should be used. The method by [9] extracts new topological features of an analog circuit using unsupervised learning techniques and generates a hierarchical building block representation of the circuit. It still relies on predefined rule-sets to be able to identify new sub-structures in a circuit. The method by [10] uses graph convolutional neural networks to cluster the devices of an analog circuit into its functional blocks, e.g., circuit biases, loads and input pairs, based on geometrical information of the circuit's devices in the schematic. It does not generate a hierarchical building block description of the circuit which could be used to identify that topology in large designs.

In the following, a new method is presented which automatically decomposes a given circuit topology into transistor or building block pairs, generates their recognition rules and consecutively assembles those pairs into larger building blocks constructing a hierarchical library with which a structure recognition algorithm is able to identify that topology.

The main contributions of this paper are:

- Algorithms to automatically generate the building block library of a given circuit which can then be used by further structure recognition. No pre-defined recognition rules are required to do so.
- A heuristic to automatically group transistor and building block pairs to larger blocks by analyzing their characteristics.
- Existing libraries can be extended by reapplying the method using a previously generated library. In this way, a library dedicated to identify a specific topology type can be obtained.
- A new building block library dedicated to level shifter topologies.

978-3-9819263-5-4/DATE21/©2021 EDAA

Algorithm 1 Automatic generation of the building block description of a circuit

```
1: procedure CIRCUITDECOMPOSITION(L_struct, C)
2:     L'_struct = L_struct
3:     repeat
4:         B = recognizeStructures(L'_struct, C)        ▷ E.g. [4]
5:         T = findTopLevelStructures(B)
6:         E_new = createNewLibraryElements(T)          ▷ Alg. 2
7:         extendLibrary(L'_struct, E_new)              ▷ Alg. 3
8:     until |T| = 1
9:     return L'_struct
10: end procedure
```

Algorithm 2 Heuristic to discover new building block pairs

```
1: procedure CREATENEWLIBRARYELEMENTS(T)
2:     E_new = ∅
3:     B' = T × T                                       ▷ Tentative building blocks
4:     B'_w = weightCharacteristics(B')
5:     for all (b_i, b_j) ∈ B'_w with descending weights do
6:         e_new = generateRecognitionRules(b_i, b_j)
7:         E_new = E_new ∪ e_new
8:         eliminateOverlappingBlocks(B'_w, b_i, b_j)
9:     end for
10:    removeDuplicateElements(E_new)
11:    return E_new
12: end procedure
```

The remainder of the paper is organized as follows: Section II describes the newly developed algorithms for library-free structure recognition. Section III presents experimental results for the new method which has been applied to different level shifter and operational amplifier topologies. Section IV concludes the paper.

II. Library Free Structure Recognition

Algorithm 1 outlines the main algorithm of the new method. The user has to provide a –possibly empty– building block library L_{struct} and a circuit netlist C as inputs to the method. It outputs the building block library L'_{struct} which enables the identification of C in a complete VLSI design by structure recognition. The library L'_{struct} is either built from scratch, i.e., when the provided L_{struct} is empty, or an extended version of L_{struct}, i.e., it is augmented by new library elements and hierarchy levels required to identify C.

Algorithm 3 Insertion of new library elements

```
1: procedure EXTENDLIBRARY(L'_struct, E_new)
2:     for all e_new = (b_i, b_j) ∈ E_new do
3:         level = maxLevel(b_i, b_j) + 1
4:         if level > maxLevel(L'_struct) then
5:             addNewHierarchyLevel(L'_struct, level)
6:         end if
7:         addElementToLevel(e_new, level)
8:     end for
9: end procedure
```

Algorithm 1 first creates the copy L'_{struct} of L_{struct}. This copy is iteratively extended by new library elements and hierarchy levels until the given circuit topology C can be identified with L'_{struct}. The algorithm identifies the building blocks B specified by L'_{struct} by structure recognition, e.g., [4]. Then, the top-level structures T, i.e., the building blocks which are not part of another building block of a higher hierarchy level yet, are extracted from B. Algorithm 2 groups these structures into pairs, generates their corresponding recognition rules, i.e., a new set of library elements E_{new}. Those elements are then inserted into L'_{struct} by Algorithm 3. This algorithm also extends L'_{struct} by a new hierarchy level, if required. In the next iteration of Algorithm 1, the structure recognition method identifies the newly added elements in the circuit, which are then used by Algorithms 2 and 3 to extend the library further. This process is repeated until T contains only one single element, i.e., the building block description of the given netlist C. Finally, Algorithm 1 returns L'_{struct}.

Algorithm 2 first initializes the set of new library elements E_{new} empty and creates the set of tentative building blocks B' by forming the Cartesian product of the top-level structures T. Those blocks are then weighted and stored in descending order in the vector B'_w. The weight of a tentative building block (b_i, b_j) is determined by analyzing certain characteristics with the goal of a structure recognition as unambiguous as possible. The following characteristics are defined:

- *Connectivity:* Pairs can be formed by sub-blocks with at least one pin connection. Unambiguous identification becomes more likely, the more connections between two sub-blocks exist. Many devices and building blocks in a circuit are connected via its supply nets, which makes the corresponding connection less meaningful compared to the internal connections between those two blocks. Hence, connections via supply nets are weighted with two points, internal connections with three points.
- *Substrate type:* Most analog building blocks are formed by two sub-blocks of the same substrate type. Hence, the substrate type weight of a tentative building block is two if the two sub-blocks have the same substrate-type, one if n- and p-type are mixed and zero for any other combination. This ensures that new building blocks are preferably formed by sub-blocks of the same substrate type. On higher levels however, sub-blocks have to be combined to mixed-type blocks.
- *Hierarchy level difference*: It is defined as the absolute value of the hierarchy level difference between two building blocks. The hierarchy level difference states that building blocks on the same hierarchy level or with a low hierarchy level difference should preferably be combined to a new building block. This ensures that building blocks on low hierarchy levels are included early into the pair assembling process.

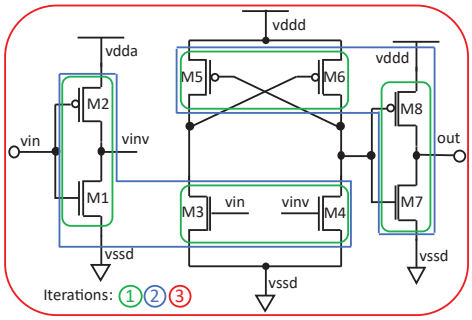

Fig. 1. Working principle of the presented method for a conventional voltage level shifter [11]

The overall weight of a tentative building block $(b_i, b_j) \in B'$ is then computed as follows:

$$\begin{aligned} weight(b_i, b_j) \; &= 2 \cdot connectivityWeight(b_i, b_j) + \\ &\quad substrateTypeWeight(b_i, b_j) - \\ &\quad hierarchyLevelDifference(b_i, b_j) \end{aligned} \tag{1}$$

As the connectivity is crucial for an unambiguous recognition, it is weighted with factor two. Please note that the weights have been heuristically chosen by hand. After weighting and storing the tentative building blocks in B'_w, Algorithm 2 iterates over them according to descending weights. Hence, new library elements are preferably built from pairings with more connections, same substrate type and closer hierarchy levels. The algorithm creates such a library element by extracting the recognition rules of (b_i, b_j) according to [4]. To avoid ambiguities due to overlapping structures, any other tentative building block which contains either b_i or b_j is eliminated from B'_w. This process is repeated until B'_w is empty. Finally, duplicate library elements which describe the same pair are removed from E_{new}. Duplicate elements might be added to E_{new} if the circuit contains non-overlapping, identically connected devices multiple times.

Algorithm 3 extends the library L'_{struct} by the newly created library elements $(b_i, b_j) \in E_{new}$. It determines the hierarchy level of each new element by incrementing the highest hierarchy level of its sub-blocks by one and adds that element to its corresponding hierarchy level. The library L'_{struct} is extended by one level if the new building block is formed by at least one sub-block on the currently highest level of the library.

Fig. 1 illustrates the working principle of the new method at the example of a conventional voltage level shifter circuit [11]. The provided initial building block library L_{struct} was empty. In the first iteration of Algorithm 1, structure recognition returns the set B containing the eight transistors of the level shifter circuit. Algorithm 2 uses these to form 20 different tentative building blocks which are then weighted and stored according to descending weights in B'_w as shown in Table I.

As an example, the transistors $(M5, M6)$ are internally connected two times (gates to drains) and once over the supply rail (sources), hence yielding a connectivity weight of $2+2\cdot3 = 8$. Furthermore, $M5$ and $M6$ have the same substrate type and lie on the same hierarchy level. Their corresponding total weight then evaluates to $2 \cdot 8 + 2 - 0 = 18$.

TABLE I
WEIGHTS OF THE TENTATIVE BUILDINGS FORMED IN THE FIRST ITERATION OF ALGORITHM 1 FOR THE CIRCUIT OF FIG. 1

$(b_i, b_j) \in B'_w$	connectivity	substrate	hierarchy level difference	total	used
$(M5, M6)$	8 $(2+2\cdot3)$	2 (p,p)	0	18	yes
$(M1, M2)$	6 $(2\cdot3)$	1 (n,p)	0	13	yes
$(M7, M8)$	6 $(2\cdot3)$	1 (n,p)	0	13	yes
$(M1, M3)$	5 $(2+3)$	2 (n,n)	0	12	no
$(M1, M4)$	5 $(2+3)$	2 (n,n)	0	12	no
$(M4, M7)$	5 $(2+3)$	2 (n,n)	0	12	no
$(M5, M8)$	5 $(2+3)$	2 (n,n)	0	12	no
$(M6, M8)$	5 $(2+3)$	2 (p,p)	0	12	no
$(M2, M3)$	3	1 (p,n)	0	7	no
$(M2, M4)$	3	1 (n,p)	0	7	no
$(M3, M5)$	3	1 (n,p)	0	7	no
$(M3, M6)$	3	1 (n,p)	0	7	no
$(M4, M5)$	3	1 (n,p)	0	7	no
$(M4, M6)$	3	1 (n,p)	0	7	no
$(M4, M8)$	3	1 (n,p)	0	7	no
$(M5, M7)$	3	1 (p,n)	0	7	no
$(M6, M7)$	3	1 (p,n)	0	7	no
$(M3, M4)$	2	2 (n,n)	0	6	yes
$(M1, M7)$	2	2 (n,n)	0	6	no
$(M3, M7)$	2	2 (n,n)	0	6	no

After weighting all 20 tentative building blocks, Algorithm 2 iterates over them in descending order. In the first iteration, the algorithm generates the recognition rules for $(M5, M6)$ and adds the corresponding library element to E_{new}. Any other tentative building block overlapping with $(M5, M6)$ is eliminated from B'_w to avoid ambiguities in the recognition process. Hence, e.g., $(M5, M8)$ and $(M6, M8)$, amongst others, are removed from B'_w.

In the second iteration, the recognition rules for $(M1, M2)$ are generated, added to E_{new} and the corresponding overlapping structures are eliminated. This step is repeated for $(M7, M8)$. Now, as all other tentative blocks are eliminated by $(M5, M6)$, $(M1, M2)$ and $(M7, M8)$, only the transistors $M3$ and $M4$ remain to be grouped. Their corresponding library element is generated and added to E_{new}.

Finally, duplicate library elements are removed from E_{new}. In this case, $(M1, M2)$ and $(M7, M8)$ describe the same structure. Hence, one of those two elements is discarded. Algorithm 2 terminates by returning E_{new}. As L'_{struct} was empty, Algorithm 3 extends L'_{struct} by one level and adds all newly created library elements to it. The transistors which have been grouped into pairs in the first iteration of Algorithm 1 are highlighted in green in Fig. 1.

In the second iteration of Algorithm 1, the pairs $[(M1, M2), (M3, M4)]$ and $[(M5, M6), (M7, M8)]$ are grouped together as highlighted in blue in Fig. 1. L'_{struct} is extended by another level containing these two elements. Afterwards, those two blocks are combined to form the conventional voltage level shifter circuit. Fig. 2 shows the final library L'_{struct} generated from scratch by the presented method. It consists of three hierarchy levels containing six device and structure pairs in total.

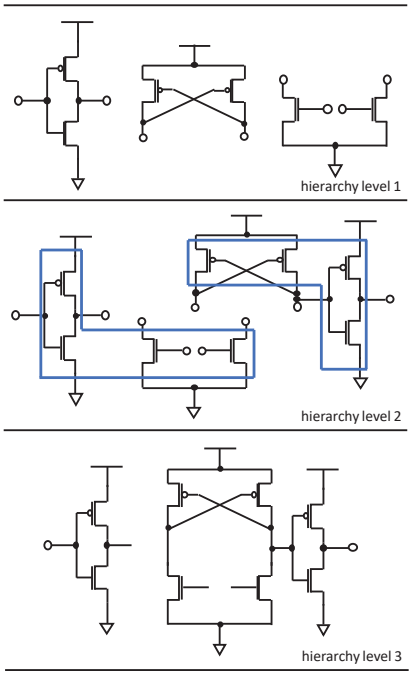

Fig. 2. Building block library L'_{struct} generated by Algorithms 1 and 2 for the conventional voltage level shifter from Fig. 1

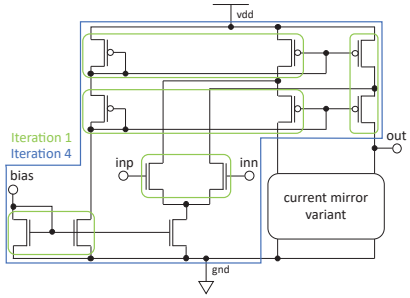

Fig. 3. Folded cascode amplifier [12]

III. Experimental Results

A. OpAmps

Fig. 3 shows a folded cascode amplifier (foca) [12]. The load in the output stage can be implemented by the different current mirror variants shown in Fig. 4. The method presented in Sec. II has been applied to the resulting six folded cascode amplifier variants. The algorithms were first executed on the foca with a cascode current mirror load and an empty initial building block library L_{struct}. The circuit bias, the differential input stage and the pmos cascode current mirror load are grouped to one building block after four iterations as highlighted in blue in Fig. 3. In the last iteration, that block and the current mirror load are combined to the building block description of the whole amplifier variant.

The generated building block library L'_{struct} has then been used on the remaining foca variants: For each variant, the blue

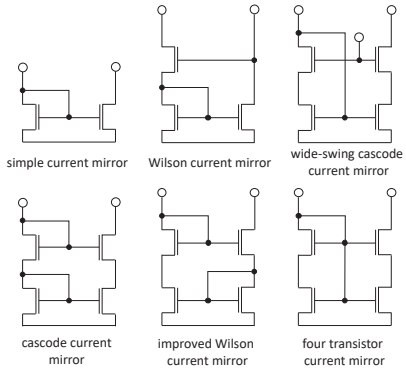

Fig. 4. Current mirror load variants

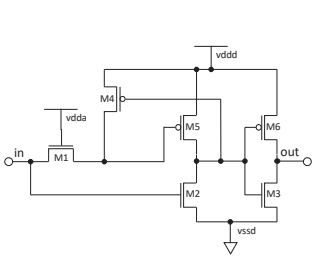

Fig. 5. Pass-gate level converter [13]

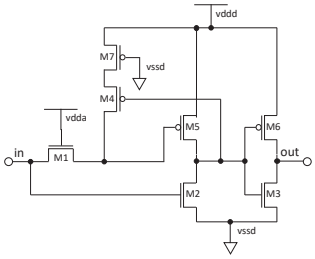

Fig. 6. Pass-gate level converter with keeper device $M7$ [13]

building block has been successfully identified. Furthermore, L'_{struct} has been extended by the respective current mirror load variant. The final building block library L'_{struct} is capable of identifying the six foca variants with a minimum number of library elements and hierarchy levels.

The presented method automatically groups the devices to differential pairs and current mirror variants by analyzing the characteristics of the formed tentative building blocks as described in Sec. II. The library L'_{struct} generated by the presented method, after analyzing only the six foca variants, contains all elements of the basic analog building block library presented by [4]. That means that the presented method was able to learn this library without prior knowledge by only the three basic rules on *connectivity*, *substrate type* and *hierarchy level difference* in Sec. II.

B. Level Shifters

Figs. 5 to 9 show different types of level shifters [11], [13]–[15]. Level shifters are commonly used in integrated circuit (IC) design to transfer signals between different voltage domains, e.g., between analog and digital parts of the IC. These circuits are susceptible to electrostatic discharge (ESD) events [8]. Hence, reliably identifying level shifters in an IC and checking whether these are sufficiently protected against ESD events by dedicated ciruitry is an important circuit verification task.

Level shifter topologies have structural similarities, e.g., the circuit from Fig. 6 complements the pass gate level converter

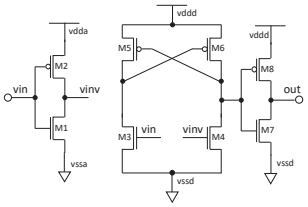

Fig. 7. Conventional voltage level shifter [11]

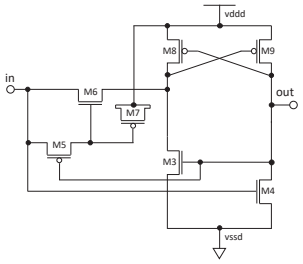

Fig. 8. Single supply level converter [14]

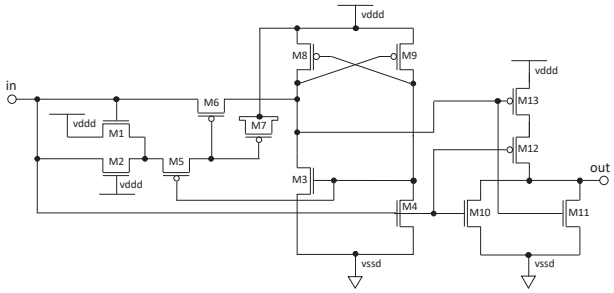

Fig. 9. Single supply true voltage level shifter [15]

from Fig. 5 by the keeper device $M7$. Hence, the presented algorithms have been first applied to the "basic" version of the pass gate level converter. The resulting library has then been extended by the additional library elements required to identify the variant with the keeper device.

This approach has also been applied to the three remaining topologies: the single supply true voltage level shifter from Fig. 9 is constructed of a NOR-gate and the single supply level converter from Fig. 8. That topology in turn contains elements of the conventional voltage level shifter from Fig. 7.

The final building block library is shown in Fig. 10. It extends the conventional voltage level shifter library of Fig. 2 by the four level converter topologies mentioned above and consists of five hierarchy levels and 20 library elements in total. Hierarchy level three contains the conventional voltage level shifter (dcvs, Fig. 7), the single supply level converter (singleSupplyLC, Fig. 8) and the pass-gate level converter (passGateLC, Fig. 5). Level four has two elements, the pass gate-level converter with keeper device (Fig. 6) and the input stage of the single supply true voltage level shifter (singleSupplyTVVS) which is connected to its output stage, an NOR-Gate, on hierarchy level five to form the circuit from Fig. 9. Level two contains the input and output stages of the level converters on level three; those stages are divided into even smaller input and output circuitry on hierarchy level one.

As shown above, our approach is capable of generating and consecutively extending a building block library dedicated to a specific circuit type, i.e., level shifter topologies.

IV. Conclusion

In this paper, a new method to automatically generate the building block library of a given circuit topology has been presented. The library can be used to identify that topology in large integrated circuit designs by structure recognition. Libraries dedicated to one specific topology type, e.g., operational amplifiers, can be obtained by applying the method to the basic version of that circuit type. The created basic libraries can then be extended by the elements required to identify additional topology variants by reapplying the presented method on those variants with the basic libraries as inputs. The implemented heuristics to group the devices and building blocks of a circuit are based on the typical characteristics of analog building blocks. Other rule sets can be used to, e.g., capture the logic function of digital blocks or to identify analog, digital and mixed signal domains of a circuit. Developing such rule sets is future work.

Acknowledgment

This work has been financially supported in part by the Deutsche Forschungsgemeinschaft (DFG) under contract number GR 1190/6-1.

References

[1] MunEDA GmbH, "WiCkeD." http://www.muneda.com/, 2020.
[2] Cadence Design Systems, Inc. https://www.cadence.com/, 2020.
[3] H. Graeb, S. Zizala, J. Eckmueller, and K. Antreich, "The sizing rules method for analog integrated circuit design," in *ICCAD*, pp. 343–349, Nov 2001.
[4] H. G. T. Massier and U. Schlichtmann, "The Sizing Rules Method for CMOS and Bipolar Analog Integrated Circuit Synthesis," *TCAD*, pp. 343–349, Nov 2008.
[5] M. Eick and H. Graeb, "A versatile structural analysis method for analog, digital and mixed-signal circuits," in *SMACD*, pp. 1–4, Sep 2012.
[6] K. Kunal, T. Dhar, M. Madhusudan, J. Poojary, A. Sharma, W. Xu, S. M. Burns, J. Hu, R. Harjani, and S. S. Sapatnekar, "Gana: Graph convolutional network based automated netlist annotation for analog circuits," in *DATE*, pp. 55–60, Mar 2020.
[7] M. Eick, M. Strasser, K. Lu, U. Schlichtmann, and H. E. Graeb, "Comprehensive generation of hierarchical placement rules for analog integrated circuits," *TCAD*, pp. 180–193, Jan 2011.
[8] H. Hung, M. Ker, S. Chen, and C. Chuang, "Abnormal esd damages occur in interface circuits between different power domains in nd-mode mm esd stress," in *IPFA*, pp. 163–166, Jul 2006.
[9] H. Li, F. Jiao, and A. Doboli, "Analog circuit topological feature extraction with unsupervised learning of new sub-structures," in *DATE*, pp. 1509–1512, Mar 2016.
[10] K. Settaluri and E. Fallon, "Fully automated analog sub-circuit clustering with graph convolutional neural networks," in *DATE*, pp. 1714–1715, Mar 2020.
[11] M. Lanuzza, P. Corsonello, and S. Perri, "Low-power level shifter for multi-supply voltage designs," *TCSII*, vol. 59, pp. 922–926, Dec 2012.
[12] P. E. Allen and D. R. Holberg, *CMOS Analog Circuit Design*. Oxford University Press, 3 ed., 2011.
[13] S. H. Kulkarni and D. Sylvester, "High performance level conversion for dual v/sub dd/ design," *TVLSI*, vol. 12, pp. 926–936, Aug 2004.
[14] Q. A. Khan, S. K. Wadhwa, and K. Misri, "A single supply level shifter for multi-voltage systems," in *VLSID*, pp. 4 pp.–, Jan 2006.
[15] R. Garg, G. Mallarapu, and S. P. Khatri, "A single-supply true voltage level shifter," in *DATE*, pp. 979–984, Apr 2008.

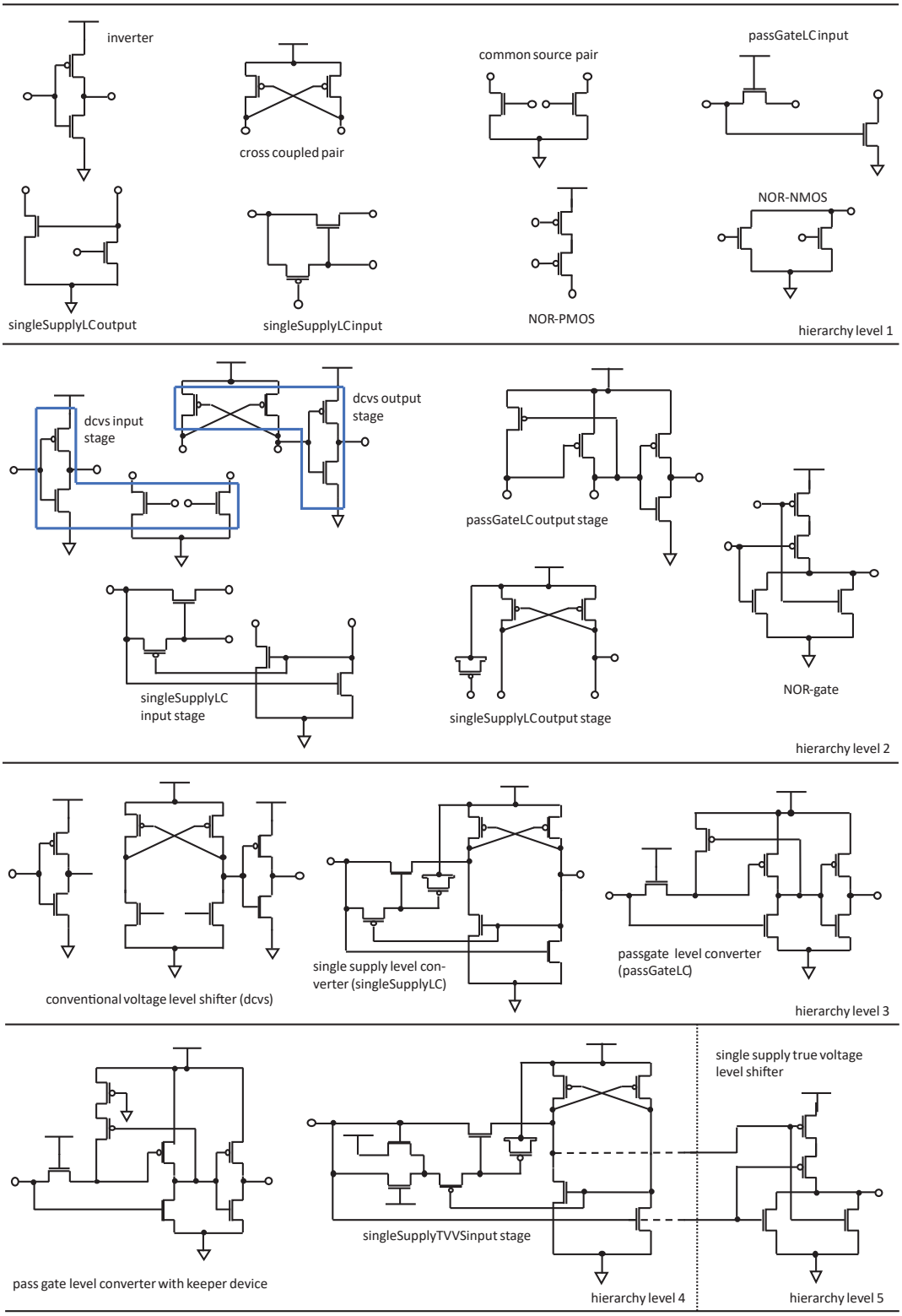

Fig. 10. Final building block library L'_{struct} dedicated to level shifter topologies