A Generic Standard Cell Design Methodology for Differential Circuit Styles

Stéphane Badel, Erdem Güleyüpoğlu, Özgür İnaç, Anna Peña Martinez, Paolo Vietti, Frank K. Gürkaynak and Yusuf Leblebici

Microelectronic Systems Laboratory (LSM) EPFL, CH-1015 Lausanne
E-mail: stephane.badel@epfl.ch

Abstract

In this paper we present a generic methodology for the rapid generation and implementation of standard cell libraries for differential circuit design styles. We demonstrate a systematic approach for the classification of circuit topologies (footprints) and for generating the templates that correspond to a large number of functions. The generation of an extensive cell library with more than 4500 standard cells based on 19 footprints is demonstrated using a 180 nm CMOS technology.

1. Introduction

The established method of designing ASICs is based on mapping the high-level description of a circuit to a library of pre-designed and characterized standard cells using synthesis tools. The result of the synthesis is a netlist of interconnected standard cells, and in a second step placement and routing tools are used to obtain the physical design. Over the years this simple design methodology has enjoyed great success mainly because it is well-suited for design automation. In a typical standard cell library, a single cell has multiple ‘views’. Each view provides different information as required by the design tools. As an example, the synthesis tool would require timing information of the cell while placement and routing tools would also require physical design information. A modern standard cell library consists of several hundred cells where each cell has as many as ten or even more views.

Standard cell libraries using standard CMOS logic are provided by several companies for all manufacturing technologies and are widely available. The same cannot be said for novel circuit realization techniques. Until their commercial viability is proven few companies find it a worthwhile investment to provide a complete standard cell library for a new circuit design technique. Some academic and research institutions have tried designing small-scale libraries (with perhaps 50 cells) but such efforts seldom result in a library that is comparable to those provided for standard CMOS design.

Differential circuits make up a relatively large family of alternative logic gate realizations to standard CMOS logic [6]. In general, differential circuits have much higher noise immunity and are inherently faster than their CMOS counterparts at the cost of larger area and increased routing overhead. Some well-known differential logic families include: DCSL [8], DCVSL [3], MCML [10], LVDCL [9], and several pass transistor logic styles [11].

There are many successful differential circuit realizations in the literature and in the industry. However most of these realizations have been designed using full-custom design methodologies or have used small scale standard cell libraries [2]. In this paper, we present a methodology for generating a large number of standard cells for differential circuits from a limited set of physical designs that we will call footprints.

While the methodology presented here is fairly general and applicable for most differential circuit styles, we will use the MOS Current-Mode Logic (MCML) as an example application [10]. In Section 2, a brief overview of the MCML circuits will be provided and the main design parameters will be explained. The methodology to design the standard cell library will be detailed in Section 3, and results will be presented in Section 4. Finally, in Section 5 conclusions will be summarized.

2. MCML logic circuits

MOS Current-mode logic (MCML) circuits have been extensively used for high-speed applications [5] and their
properties are well studied [1]. A simple MCML gate realizing the 2-input XOR function is shown in Figure 1. The MCML gate operates with a constant tail current $I_{\text{bias}}$. The nMOS network realizes the Boolean logic and steers the tail current to one of the two output load resistors resulting in a voltage drop at one of the output terminals. As a result, MCML gates have constant power consumption, a property which makes them an interesting alternative for cryptographic and EMI sensitive applications [7].

2.1. Design parameters

The main parameter that determines the performance of an MCML gate is the bias current $I_{\text{bias}}$. Higher bias currents result in faster operating speeds at the expense of increased power consumption. The speed-up obtained in this way is linear for a wide range of the bias current allowing excellent trade-off between operating speed and power consumption.

Once the bias current has been determined the output logic swing ($V_{\text{swing}}$) can be determined by sizing the output load resistors appropriately. In simple terms, larger values of $V_{\text{swing}}$ results in more robust circuits at the expense of operating speed. Thus, the output delay of the MCML gate is determined by

$$\tau_p = \frac{C_{\text{load}} \cdot V_{\text{swing}}}{I_{\text{bias}}} \quad (1)$$

The logic function is realized by a network consisting of multiple levels of nMOS transistors, which is determined by the binary decision diagram (BDD) corresponding to the Boolean function being realized. The method used to determine the specific network used by each MCML gate will be explained briefly in Section 3. The transistors in each level of the gate must be sized to ensure a certain noise margin (NM). As a simple rule, transistors in the lower levels of the network will be larger than those in higher levels. In practice, up to 3 levels of transistors are sufficient to generate a very comprehensive standard cell library of several thousand cells.

For a family of MCML based standard cells it is sufficient to determine the three parameters, $I_{\text{bias}}, V_{\text{swing}}$ and NM. The above mentioned guidelines give a good starting point for proper sizing. However, a more detailed analysis is required to design optimized netlists as the NM also depends on the transistor sizes, and as a result for a given output load it is possible to determine the optimal $V_{\text{swing}}$ for given NM and $I_{\text{bias}}$.

One common problem in standard cell libraries is to design cells with different drive strengths. For MCML gates, the solution is to simply increase $I_{\text{bias}}$. To preserve the same $V_{\text{swing}}$, the resistance of the pMOS load device has to be increased by the same amount as well. Finally, to have the same NM all transistors within the logic network have to be scaled as well.

Figure 1 shows the transistor level schematic of an MCML gate that realizes the 2-input XOR function. Transistor dimensions are given for a 180 nm technology with $I_{\text{bias}} = 20 \mu A$, $V_{\text{swing}} = 320 mV$ and a NM of 30%. Note that two DC bias voltages $V_n$ and $V_p$ are required for the bias current and the output load respectively.

2.2. Differential Routing Issues

One of the challenges in developing a design flow for a differential circuit is back-end design. Modern placement and routing tools are unable to recognize complementary outputs, and by default will route each output pin separately. This results in a sub-par layout where differential signals are not routed side by side. This has two main drawbacks. First, the crosstalk immunity is adversely affected since the two complementary wires do not receive equal noise contributions. Secondly, the output capacitive load is no longer equally balanced between the two output pins resulting in switching noise.

A simple method to achieve true-differential routing is to use the so-called fat wires. In this approach, a pair of differential pins are merged into a larger single ended pin. The router will then connect these pins using a fat wire. At this point the fat wire and the fat pin will be split into a pair of differential wires and pins respectively. The

![Figure 1. Transistor level schematic of an MCML gate realizing the 2-input XOR function](attachment:figure1.png)
3. Designing MCML Standard Cells

Once the electrical parameters and transistor sizes for individual cells have been determined it is a trivial but onerous task to design all components of a standard cell library. Depending on the quality of the standard cell library for hundreds of cells, transistor level schematics have to be designed, their timing characteristics have to be extracted, and a corresponding physical layout has to be prepared. Furthermore, every time a new manufacturing technology is targeted this time consuming process has to be repeated.

There are some important properties of differential circuits that can be exploited to significantly accelerate the design process. Here, we will present a methodology where starting from less than 20 physical layouts we will automatically generate in excess of 4,500 standard cells. This methodology has three distinct steps. First, different footprints for the network of nMOS transistors are created. Each of these footprints represents a specific network of transistors and has a unique physical layout. In the second step, standard cell templates are created by assigning the transistors of the nMOS network to the inputs of the logic gate. The last step takes advantage of the fact that for differential cells, swapping the pin connections of a differential pin results in an inversion at that pin. This inversion is for all practical purposes for free. By inverting the inputs and output of standard cell template a large number of standard cells implementing different logic functions can be generated while retaining the same physical layout (template).

3.1. Footprints

The core of a differential cell is the network of nMOS transistors. This network can be represented using a binary decision diagram (BDD) where each node of the BDD is a differential pair and each branch of the BDD is a connection between one drain and the source of another differential pair or an output as shown in Figure 4. The square zero and one nodes represents the complementary outputs of the MCML gate. The BDD is a very practical way to capture the be-

---

3. Physical Layout

Two standard cell layouts for MCML-based circuits are given in Figure 3. The DC bias voltages $V_n$ and $V_p$, which are common for all gates in the design are routed parallel to supply lines on the same layer. The current mirror for the bias current is placed underneath the ground rail, while similarly the pMOS load transistors are placed under the supply line. The standard cell height is mainly determined by the nMOS network.

---

2.3. Physical Layout

Two standard cell layouts for MCML-based circuits are given in Figure 3. The DC bias voltages $V_n$ and $V_p$ which are common for all gates in the design are routed parallel to supply lines on the same layer. The current mirror for the bias current is placed underneath the ground rail, while similarly the pMOS load transistors are placed under the supply line. The standard cell height is mainly determined by the nMOS network.
Figure 4. Six out of the nineteen footprints considered for MCML BDD topologies.

havior of the MCML, which steers the bias current to one of the outputs depending on the inputs.

Note that there are only a limited set of possible footprints, given the number of input variables. For example the first three footprints (#01, #02,#03) are sufficient for all networks with 1- and 2-levels of differential pairs.

Increasing the number of levels in the logic network allows more complex functions to be realized within a single gate. However, each additional level influences the switching speed of the gate. In an MCML network, the differential pairs closest to the output nodes are always the fastest to switch since they need to drive less parasitics than the differential pairs at lower levels. In our work, we consider gates up to 3 levels, and by performing an exhaustive search we have determined that 19 footprints will be sufficient to implement all logic functions that can be mapped to MCML gates with up to 3-levels.

3.2. Generating Templates

The most complex function that can be realized with a network of $N$ levels is the $2^{N-1}$-to-1 multiplexer. Such a gate will be able to realize all possible functions of $N$ variables, by properly assigning the polarity of the different inputs. In addition, it can also implement a number of functions up to $2^N - 1$ variables.

In our case, for 19 different footprints we can implement 63 unique functions with 1 to 7 input variables. In addition, there are 45 functions that implement the same function in an alternate way. While these additional functions seem redundant from a logical point of view, they have different electrical characteristics because they are implemented using different physical layouts. As a result, a total of 108 standard cell templates are generated by different assignments of the MCML inputs to the 19 footprints.

3.3. Standard Cells

As mentioned earlier, inversions of the input and output signals can be done by simply swapping the complementary pins in differential circuits. This allows a large number of logic functions to be generated from a single template. As an example, the template that implements the 2-input AND gate can be used to generate seven other logic functions, including 2-input NAND, NOR, and OR functions.

From the 108 standard cell templates, it is possible to generate 4660 standard cells by inverting the inputs and outputs. Similar to the situation with the templates some of these cells are equivalent. In some cases these redundant cells have different electrical characteristics, while some others are truly redundant. Table 1 presents an overview of all possible logic functions.

Note that, for all these 4660 cells there are only 19 physical layouts. In the first stage the physical layouts of the 19 footprints are generated. Different drive strength versions of these layouts are obtained by scaling the footprints.

<table>
<thead>
<tr>
<th>Inputs</th>
<th>functions</th>
<th>redundant</th>
<th>I/O variants</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>0</td>
<td>16</td>
</tr>
<tr>
<td>3</td>
<td>20</td>
<td>10</td>
<td>320</td>
</tr>
<tr>
<td>4</td>
<td>53</td>
<td>29</td>
<td>1696</td>
</tr>
<tr>
<td>5</td>
<td>25</td>
<td>6</td>
<td>1600</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>0</td>
<td>768</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>0</td>
<td>256</td>
</tr>
<tr>
<td>Total</td>
<td>108</td>
<td>45</td>
<td>4660</td>
</tr>
</tbody>
</table>

Table 1. Number of logic functions realized in the library, per number of inputs
Figure 5. Mapping file for generating templates

| F1 | T1_buffer | A | Y | "Buffer" |
| F1 | T1_inverter | A | nY | "Inverter" |
| F2 | T2_and2 | A | B | Y | "And2" |
| F2 | T2_nand2 | A | B | nY | "Nand2" |
| F2 | T2_or2 | nA | nB | nY | "Or2" |
| F2 | T2_nor2 | nA | nB | Y | "Nor2" |
| F3 | T2_xor2 | A | nB | B | Y | "Xor2" |
| F3 | T2_xnor2 | A | nB | B | nY | "Xnor2" |
| F3 | T2_mux2 | S | In1 | In0 | Y | "Mux2" |

Standard cell templates are automatically generated from the footprints by using a mapping file as shown in Figure 5. In this file the first column specifies the footprint to be used, the second column is the name of the template. The next columns form an ordered list of pin assignments to the differential pairs of the template. The special prefix ‘n’ is used to signal an inversion at that pin location. As an example the second to last line describes the 2-input XNOR function. It uses the footprint number 3 (shown in Figure 4). The input A is mapped to the differential pair AO, and B is mapped to two differential pairs A1 and A2 whereby the connection to A1 is inverted. The last pin is the output of the gate, and this is inverted as well. The last entry of the row is optional and is used to select a standard symbol for the template.

At this point the standard cell templates can be characterized for timing. Note that the cells generated by I/O inversions have different logic functions, but they inherit the timing information from the standard cell templates.

For placement and routing based on the initial physical layouts of footprints two separate libraries are automatically generated. The first library is the so called far library which contains a single-ended equivalent of the cell, and the second library contains the real differential pins. Both layouts are generated automatically from the footprints and their pin information is updated according to the standard cell definition. The end result is the complete library with views for transistor level simulation (schematic), schematic capture (symbol), synthesis (lib), placement and routing (abstract), and final physical design (layout) for more than 4,000 standard cells for a single drive strength.

3.4. Sequential Cells

Sequential cells are generated by using the same footprints. To create the latch functionality, a multiplexer cell is used in the feedback configuration. Flip-flops are created from latches in master-slave configuration. The physical layouts of these cells are optimized separately. Flip-flops with embedded functions (enable, scan etc) can be generated using only two footprints. The optimum combination of cells for different drive strengths are obtained by extensive simulations.

4. Results

The standard cell library developed by the methodology presented in this paper has been used in the differential design flow of various benchmark circuits. Among other designs we have implemented a 2.5 Gb/s Advanced Encryption Standard (AES) chip of medium complexity of about 40k gate equivalents using a standard 180 nm technology. Figure 6 shows a close up of the AES design where the differential routing is clearly visible.

The generated differential standard cells have timing characteristics comparable with conventional CMOS equivalents, while providing excellent noise immunity properties. As mentioned earlier, different standard cells can be obtained by inverting the inputs and outputs, creating a large number of cells. We have mapped several designs using the automatically generated MCML library to see how well these variants can be utilized using standard synthesis tools. Figure 7 shows the distribution of input/output variants of a single standard cell template called MA21V1 within the synthesized netlist of the AES design mentioned earlier. It can be seen that most variants of the standard cell have been
utilized. The additional variants indeed enrich the standard cell library and allow tools more freedom for optimization. Finally, Figure 8 shows the power supply noise generated by the MCML design, compared to the noise generated by the equivalent CMOS circuit. The improvement is typically in the order of 40 dB.

5. Conclusions

Rich standard cell libraries are essential for high quality digital designs. While standard CMOS based standard cell libraries are widely available, especially for new technologies such as current mode differential logic to best of our knowledge there are no available standard cell libraries. In this paper we present a design methodology that allows us to create a rich standard cell library for differential logic styles. Based on only 19 physical layouts, we can automatically generate an extensive standard cell library with more than 4,500 standard cells including all necessary design views for standard EDA tools. Although the methodology is presented based on the MCML logic style, it is generic enough to be employed for all differential logic styles.

References