A Language to Design Generators of Analog Functions

P. Nguyen Tuong, V. Bourguet, L. de Lamarre, M.-M. Louërat, A. Greiner
University Pierre et Marie Curie, LIP6/ASIM Laboratory
4 Place Jussieu, 75005 Paris, France

Abstract

Here we present the CAIRO+ language, based on C++ functions, dedicated to the design of analog function generators. This language is aimed to enable the designer to store his knowledge into a generator. The generator is used to size fixed topologies with different specifications. It can also be used to migrate analog functions between different technologies.

1 Motivation

This work tries to handle analog circuits that require, for the same electrical topology, frequent electrical re-sizing to meet new specifications as well as thorough layout re-shaping. This is the case of commonly used functions that are useful in many different contexts. Such functions have to face new specifications, or even more difficult, they have to migrate on a new technology process. Specification and technology migration is a hot topic in analog and mixed design today and there is still a lack of methodology and tool to solve the related problems. However, it is compulsory to find how to re-use existing design and knowledge, and to define and use some kind of analog libraries of analog functions.

In order to convince the analog designer to store his knowledge for re-use, we are introducing the CAIRO+ methodology and language. Analog circuits are considered as a hierarchy of devices and modules. The module tree is used to represent the netlist template of the circuit, where the leaf cell of the electrical hierarchy corresponds to a device. Devices consist of elementary components such as folded MOS transistors, capacitors and resistances but also sets of elementary components that have to be matched (i.e. differential pair, current mirrors, capacitor matrices, matched resistances). A mechanism, which is established between levels of the hierarchy, allows modules to question devices about their electrical behavior (fig. 1-b).

CAIRO+ allows close interaction between electrical sizing and layout realization. It follows a layout educated electrical sizing approach, based on fixed layout templates and relying on layout device generators [DLP00, NLG04]. The container tree is used to describe the relative placement of instances inside a module, where the leaf cells of this tree correspond to devices.

As CAIRO+ is a language, the designer is helped to capture his tuning expertise inside a generator. This generator is used to size the fixed topology for different specifications. The same generator can be used to migrate the same topology between different technologies (fig. 1-a).

2 Generator Design Example

To capture his expertise inside a generator, CAIRO+ provides the designer a library of functions that are related to each step of the flow presented on Fig. 1 : netlist template, layout template, specification template, design space exploration, layout shape computations and layout generation.

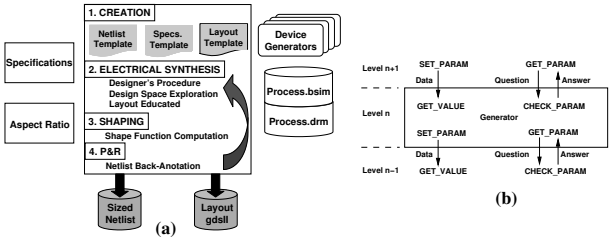

Figure 1: CAIRO+ Analog generator (a) – Communication Mechanism (b)

Here follows an example featuring a part of an OTA generator sizing procedure. The differential pair sizing is called, providing transistor width w_dp and small signal parameters in order to compute OTA gain-bandwidth GBW.

```
CAIRO_BEGIN_SIZE(OTA)
CAIRO_BEGIN_CHECK_PARAM("GBW")
  CAIRO_TRY_GET_VALUE("IBIAS_OTA",ibias_ota)
    IF_NO_VALUE
      FATAL_ERROR_PARAM("IBIAS_OTA","parameter not set",LOCATION);
    ENDIF_NO_VALUE
  // compute differential pair DP bias current from OTA bias current
  ids_dp = -ibias_ota;
  CAIRO_SET_PARAM("libMOS","DP_MOS","DP","VGS",vgs_dp) ;
  CAIRO_SET_PARAM("libMOS","DP_MOS","DP","VDS",vds_dp) ;
  CAIRO_SET_PARAM("libMOS","DP_MOS","DP","VBS",vbs_dp) ;
  CAIRO_SET_PARAM("libMOS","DP_MOS","DP","IBIAS",ids_dp) ;
  CAIRO_SET_PARAM("libMOS","DP_MOS","DP","L",l_dp) ;
  // call differential pair sizing procedure
  TRY
    CAIRO_GET_PARAM("libMOS","DP_MOS","DP","W","W(L,IBIAS,VGS)",w_dp);
    CAIRO_GET_PARAM("libMOS","DP_MOS","DP","CGD","*",cgd_dp);
    CAIRO_GET_PARAM("libMOS","DP_MOS","DP","CBD","*",cdb_dp);
    CAIRO_GET_PARAM("libMOS","DP_MOS","DP","GM","GM(W,L,VGS)",gm_dp);
  IF_ERROR_PARAM
    exit(1) ;
  ENDIF_ERROR_PARAM
  // compute OTA gain-bandwidth
  capa_ota = cgd_cm + cdb_cm + cgd_dp + cdb_dp + capa_load;
  gbw_ota = gm_dp/capa_ota;
  CAIRO_RETURN_PARAM(gbw_ota);
END_CHECK_PARAM
CAIRO_END_SIZE(OTA)
```

3 Conclusion

This ongoing study is developing the CAIRO+ language, helping the designer to capture his expertise into an analog function generator, enabling close interaction between electrical sizes and layout dimensions.

References

[DLP00] Dessouky, Louërat, and Porte. Layout-Oriented Synthesis of High Performance Analog Circuits. *Design Automation and Test in Europe, DATE*, pages 53–57, March 2000.

[NLG04] Nguyen Tuong, Louërat, and Greiner. Managing the Shape Function of Analog Devices in a Slicing Tree Floorplan . *Proc. of the 11th Int. Conf. on Mixed Design of Integrated Circuit and Systems, (MIXDES)*, pages 226–229, June 2004.