Analogue Layout Generation By World Wide Web Server-based Agents

L.T. Walczowski, D. Nalbantis, W.A.J. Waller & K. Shi

Electronic Engineering Laboratory, University of Kent, Canterbury, Kent CT2 7NT, U.K.

E-mail: L.T.Walczowski@ukc.ac.uk

Abstract

A World Wide Web (WWW) based client/server system has been developed which allows server-side process independent layout generators to generate the design rule correct geometry of analogue components such as resistors, capacitors and transistors for a design system running on a local workstation. The complete system is based on the bidirectional interface between a WWW browser and a VLSI design system, with layout generators running remotely on a WWW server.

Introduction

One of the principal features of analogue full-custom layout design is that libraries of standard cells cannot be used because both the device sizes and the functionality of the analogue cells vary greatly from design to design. To compensate for this, leaf-cell layout generators are playing an increasingly important role in analogue design automation, where they assist the designer by automatically generating design rule correct mask level geometry for the leaf cells. Such generators use as their input a number of parameters that characterize the leaf-cell to be synthesized and produce the full-custom layout for that cell.

To be useful, custom layout leaf-cell generators should be able to produce the most commonly used analogue circuit components such as resistors, capacitors and transistors and the most common forms of those cells in cases where more than one realization is possible. Furthermore, these generators should be process-independent so that as technologies advance the same generator can be used without alteration.

A library of custom layout generators can be used both by an analogue IC layout synthesis system, and for manual layout construction by a designer. In the former case, the layout synthesis system calls the generators from the library to generate the leaf-cells of the design and, once the leaf-cells are ready, assembles the system by placement and routing procedures. In the latter case, the designer invokes the generators through an appropriate user interface, and then manually places and routes the generated components.

Layout generators for analogue components are provided as part of either a design system or a technology design kit from a foundry. In both cases they are platform dependent and suffer the same disadvantages as the full design system in terms of support and distribution. This paper describes the client-server technology utilized at Kent which solves these problems. The technology allows World Wide Web (WWW) based layout generators, originally developed as part of an analogue synthesis system[1], to generate layout remotely for a design system running on the local work-station. The remote generators directly control the design system on the local work station.

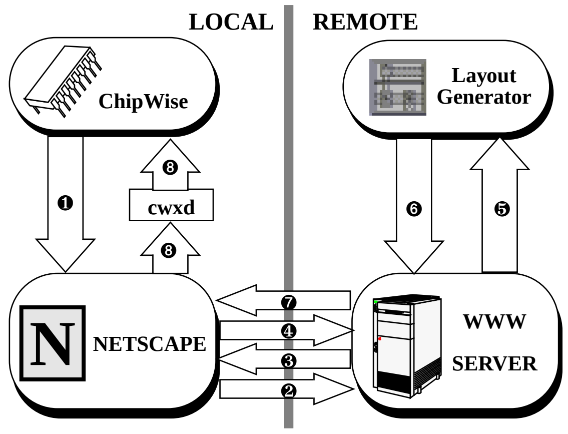
The advantages of this technology compared with conventional workstation hosted design techniques are many. Not only are the program binaries developed for a single platform, that on which the WWW server software runs, but also new versions of the layout generators are immediately available to all users. Furthermore, bugs when discovered can be fixed quickly and these fixes made available to all users immediately. Users no longer have to wait for the next release of the design system. Security issues are addressed using WWW based security mechanisms such as password protection to restrict access to authorized users or sites. Unauthorised copies of the software cannot be made, since the tools reside on the server. Extension of this technology from just the server resident layout generators would see the design system which runs locally augmented by specialised, technology specific design tools hosted on servers. The integrated WWW form based user interface provides an ideal interface for these tools with integrated help pages available through the WWW.

The system developed at Kent is called Analog Chip-Wise, and is based around the bidirectional interface of the WWW browser, Netscape Navigator[2], to the VLSI design system ChipWise[3]. However, the technology would be as easy to use with an alternative browser and a different design system. With Analog ChipWise, the user runs the conventional design system on the local UNIX workstation, whilst the WWW browser provides a form-based interface through which parameters are set for server-side layout generators. These generators, running on the WWW server[4], generate the appropriate commands which control the local design system creating design rule correct geometry for the selected analogue component.

System structure

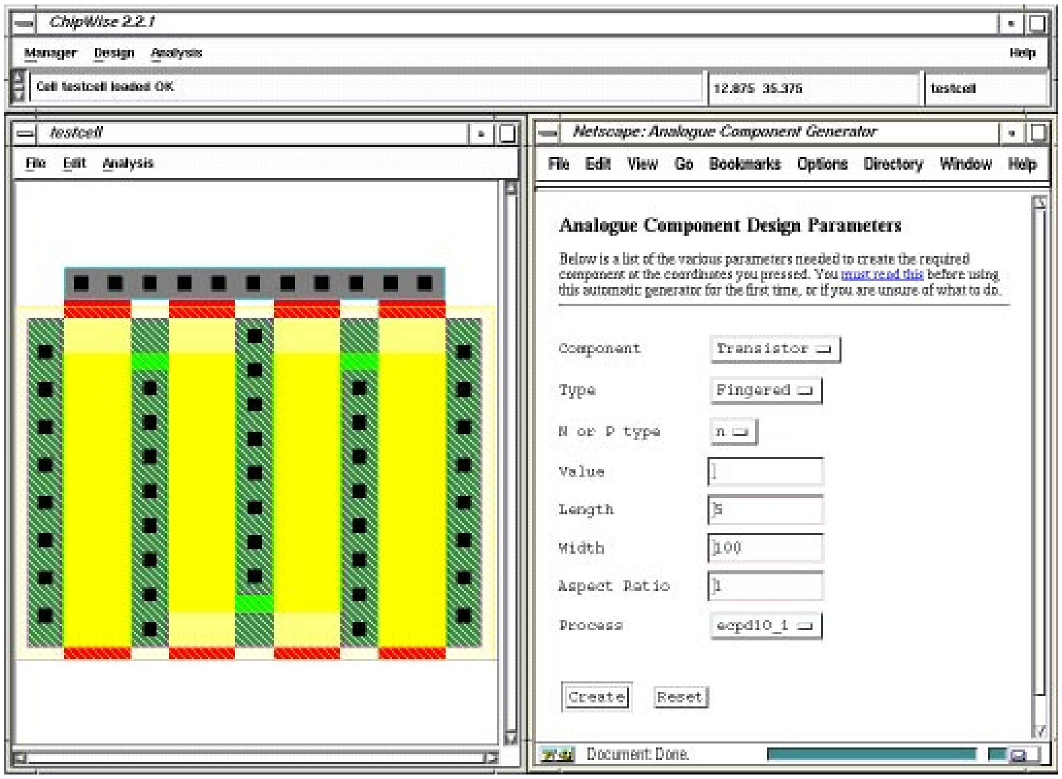
Figure 1 illustrates the major components of Analog ChipWise. The links between individual modules show the communication path whilst a request for an analogue component is handled by program modules on the local and remote systems.

A button press within ChipWise generates a Netscape remote control command to open the *Analogue Component Generator* WWW page. The coordinates of the button push are embedded as part of the URL (step 1). This request is forwarded to the WWW server (step 2), which results in the form to select the layout generator being returned (step 3). Once the required component has been selected, the completed form is sent to the WWW server (step 4). This form is interpreted by the PHP/FI program[5], which runs the required layout generator as a system call on the remote server (step 5). The layout



❶ Coordinate passed as part of URL
❷ Open URL request
❸ Form to select layout generator returned
❹ Completed form with parameters set passed to server
❺ Required layout generator run as system call
❻ CWX commands returned
❼ CWX commands sent to local host
❽ CWX commands passed to design system

FIGURE 1 Structure of Analog ChipWise


ChipWise 2.2.1
Manager Design Analysis
Help
testcell
File Edit Analysis
Netscape: Analogue Component Generator
File Edit View Go Bookmarks Options Directory Window Help
Create
Reset

Cell testcell loaded OK

12.875 35.375

testcell

Analogue Component Design Parameters

Below is a list of the various parameters needed to create the required component at the coordinates you pressed. You must read this before using this automatic generator for the first time, or if you are unsure of what to do.

Component	Transistor
Type	Fingered
N or P type	n
Value	
Length	5
Width	100
Aspect Ratio	1
Process	ecpd10_1

Document: Done.

FIGURE 2 Using Analog ChipWise to generate a 100u x 5u n-type transistor.

generator outputs commands (step 6) which are returned to the local system (step 7), in a new command language called `cwx`. This new language was developed to allow ChipWise to be controlled by a remote application (in this case the layout generator). The key module allowing Netscape Navigator to control ChipWise is the `cwxd` program which passes `cwx` commands to ChipWise whenever these commands are received from the WWW server (step 8).

Figure 2 shows Analog ChipWise in use. The top of the window is the standard ChipWise interface. The user clicks the Help item on the main menu-bar to invoke the Netscape browser and then selects the generator option from the ChipWise user interface. A button press in the edit window (lower left of Figure 2) sets the lower left coordinate of the required analogue component and calls up the *Analogue Component Generator* WWW page (lower right of Figure 2). Having completed the form and pressed the “Create” button, the user sees the required geometry generated automatically in the edit window. In Figure 2, a 100 micron wide, n type transistor has been created for ES2’s 1 micron process.

System implementation

Remote control of Netscape

On the local system, the bidirectional communication between ChipWise and Netscape is based on X properties. Appropriate WWW pages are requested using the Netscape remote control mechanism[6]. ChipWise communicates with Netscape using the same X properties that have already been registered by Netscape when the browser is invoked. The command to open the *Analogue Component Generator* home page is thus simply appended to the “_MOZILLA_COMMAND” property.

The ability to open URLs directly from the design system allows coordinates to be passed to the layout generators. For example, the coordinates of a button press in a ChipWise drawing window (e.g. x =1000, y = 800), can be added to the end of a URL, separated from the regular URL by a question mark, e.g.

```
http://eleceng.ukc.ac.uk/htbin/php.cgi/
chipwise/aladin/design.html?1000+800
```

The URL, in this case, refers to the *Analogue Component Generator* page called design.html. The coordinates are subsequently extracted and processed as necessary.

Form based user interface to a layout generator

The forms interface in HTML documents provides a powerful method for developing very user friendly, interactive applications. The *Analogue Component Generator* form, after completion by the user, is *posted* to the *generator.html* page located on the WWW server. The usual way of processing forms on WWW servers is by using Common Gateway Interface (CGI) programs. However with Analog ChipWise, the forms are processed with the PHP/FI package[5]. PHP/FI is a simple programming language embedded within HTML files, which eliminates the need for numerous CGI programs by allowing simple scripts to be placed directly into HTML files. The *generator.html* page is thus processed by the PHP/FI program, with variables substituted and values set from the posted form. *generator.html* also contains a system call to run the appropriate layout generator. For example, the PHP/FI code segment from *generator.html*, which determines whether the required transistor is to be straight, folded or fingered, is reproduced below.

```
<?if (reg_match(“Straight”,”$type”));
    $config = 0;
  elseif (reg_match(“Folded”,”$type”));
    $config = 1;
  else;
    $config = 2;
endif>
```

The cwx command language

The ChipWise design system contains design tools such as symbolic and mask level editors and analysis tools such as simulators, circuit extractors and design rule checkers. A simple, easy to parse, easy to extend command language called `cwx` was developed which could operate these tools. ChipWise command files in `cwx` are generated by the layout generators. As an example, the simple `cwx` script which creates a 10K resistor for ES2’s 1 micron technology at the origin is:

```
mask
message Making straight 10000 Ohm resistor
layer poly
box 0 0 800000 2000
layer cut
box 600 600 800 800
layer m1
box 0 0 2000 2000
layer cut
box 798600 600 800 800
layer m1
box 798000 0 2000 2000
```

Normally these command files would simply be transferred to the user’s workstation as ascii files. However all `cwx` files are interpreted by the ChipWise Web server as a new Multipurpose Internet Mail Extensions (MIME) type[7], and passed as such to the local workstation. Each user of the system has a modified .mailcap file which adds a line similar to that reproduced below for the Silicon Graphics platform:

```
chipwise/x-cwx; /usr/chipwisex/bin.SGI/cwxd %s
```

This instructs the application on receiving a cwx file to run the cwxd program automatically on the received file, in precisely the same way as a .wav audio file is passed to an audio player, or a .ps postscript file is passed to an appropriate viewer. This mechanism was chosen, rather than the more obvious use of shell scripts, to circumvent the very serious but well known security problems inherent in trying to run shell scripts remotely through the web. This problem is well documented[8].

The cwxd program

The cwxd program is invoked automatically whenever a cwx file is received from the WWW server. Its sole function is to transmit the cwx file directly to ChipWise using X properties. On invocation, cwxd transmits an *atom*, which has been defined for both itself and the ChipWise design system, to the local X server searching for a ChipWise application. If none exists, cwxd returns the message "chipwise not running on local display" and exits. If a ChipWise application, displaying on the local display is found, the cwx commands are passed to that application.

The layout generators

The layout generator system consists of three basic components: a technology database, the generator agents, and an interface for passing parameters to the agents. The technology database contains the technology parameters that are needed by the generator agents. Since CMOS has become the dominant IC fabrication technology for mainstream applications, only CMOS processes are supported. Each generator agent, implemented in C++, is responsible for the process-independent generation, in cwx format, of a specific analog component such as transistor, resistor, or capacitor. The user interface used at present for passing parameters to the generators is shown in Figure 2.

There are four main processing phases for each type of generator agent: technology loading, specification parameter evaluation, geometry generation and output formatting. Failure to achieve the goals during any of these phases results in the rest of the sequence being aborted.

During technology loading all the design rules for the specified process are loaded into the generator's run-time memory for fastest access during the geometry generation phase. Any errors during this phase e.g. missing process files, unknown parameters, etc. will cause the generator to abort its processing and exit with an appropriate error message.

After loading the process data, the generator enters the specification parameter evaluation phase in which the generator parameters specified by the user are examined and their feasibility is assessed. There are two types of specification parameters: fixed and variable. Fixed specification parameters are those whose values have a direct contribution to the electrical characteristics of the module being generated, e.g. the width and the length of a transistor, and as a result, the corresponding specified values have to be achieved at any cost. Variable specification parameters are those whose values do not have a direct contribution to the electrical characteristics of the module being generated, e.g. the aspect ratio of the transistor cell, and as a result the corresponding specified values do not have to be achieved with the specified accuracy. If a fixed parameter cannot be achieved, the generator decides at run-time the action to be taken. For example, if the specified width is less than the minimum possible width for the process the generator will use the minimum width and will notify the user with an appropriate message. If a variable specification parameter cannot be achieved exactly, the generator tries to achieve the closest possible value while keeping the fixed parameter value constant.

In the geometry generation phase each generator agent produces the masks for the cell based on the cell characteristics determined during the specification parameter evaluation phase. Although the details of mask generation are particular to the type of module being generated, all geometry is generated with a certain orientation, which is specific to each type of component, and is positioned such that the lower left point of the respective bounding box is at the origin of the orthogonal axes. It is up to the client of the generators to alter the orientation and/or the position of the generated cells.

In the output formatting phase the generator agent translates the generated masks into cwx format.

Transistor geometry generation

The transistor configurations that are supported are single-gate, multi-gate (fingered), and folded-gate. The need for multi-gate and folded-gate transistors arises from the fact that these devices in analogue circuits tend to have large widths which results in very narrow and long aspect ratios for the generated cells. This in turn not only makes it difficult to arrange such transistors on silicon in an area efficient manner but also makes it difficult to match the characteristics of such transistor pairs as required by analogue IC design. The number of gates and folds for multi-gate and folded-gate transistors respectively is decided by the generator based on the required width, length, aspect ratio and fabrication process. In addition, in order to reduce the resistivity and hence increase the conductivity of source/drain and gate regions, as many contacts as possible are generated for these regions and they are covered with metal across their entire length.

Resistor geometry generation

The resistor configurations that are supported are straight and folded. Although the folded configuration seems to be the configuration of choice for many cases due to the low-resistivity of the fabrication masks for most processes, there are also processes which supply special high-resistivity masks for resistor implementation. As a result, small resistor values with a folded configuration on such high-resistivity masks cannot be achieved and a straight resistor configuration becomes necessary. To compensate for edge-roughness effects, which alter the predicted resistance of the generated cell, both types of resistor generators use mask widths which are double the minimum width specified by the process.

Capacitor geometry generation

The capacitor configurations that are supported are the standard two plate capacitor and the three plate capacitor configurations. For both types of capacitor, the generator takes into account both the plate-to-plate capacitance and the perimeter capacitance specified by the fabrication process. Furthermore, the effective area of the capacitor is always determined by the top plate which is made slightly smaller than the other plates to compensate for inaccuracies in the capacitance value that might be introduced by mask misalignments if all masks had the same dimensions.

Results

Figure 3 shows the mask level layout of an operational amplifier designed for the MIETEC foundry. Each of the individual components in Figure 3 was generated with Analog ChipWise. The components were subsequently interconnected manually. The overall time taken to lay out the complete op-amp was less than one hour compared to several days had the whole circuit been laid out manually.

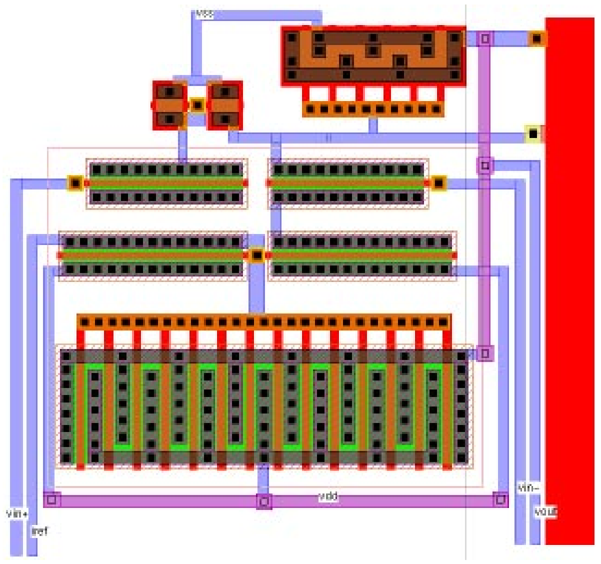


FIGURE 3 Op Amp design for the MIETEC foundry laid out with Analog ChipWise

Conclusions

A set of process independent layout generators has been developed which run as World Wide Web server-side agents and generate the design rule correct layout of analogue components such as resistors, capacitors and transistors. Each layout generator runs efficiently as compiled object code on the WWW server. The only overhead with this technology is the file transfer time between the remote and local systems. However, because these files are small in size, the duration of these transfers is very short and in practice, provided the WWW server is not overloaded, the user sees the generated layout appear in a ChipWise window almost immediately after a request has been invoked. Furthermore, because these are WWW based tools, they do not suffer from the usual disadvantages of a conventional design system running on the local workstation. The technology developed for this system could easily be applied to generate layout for other design systems. For example, the server-side layout generators could control the Cadence design system running on the local workstation, by outputing SKILL commands.

References

[1] *Rapid Layout Synthesis for Analog VLSI,* L.T. Walczowski, W.A.J. Waller, D. Nalbantis & K.Shi, 1996 IEEE International Conference on Electronics, Circuits, and Systems, Rodos, Greece, Oct. 13th -16th 1996.

[2] Netscape Navigator WWW browser, URL - http://home.netscape.com.

[3] Teaching Full-Custom Design Skills using ChipWise, L.T.Walczowski, W.A.J.Waller and E.Wells, Proc. IEE Part G, April 1992.

[4] Electronics Resources WWW server, URL - http://eleceng.ukc.ac.uk.

[5] PHP/FI home page, Rasmus Lerdorf, URL - http://www.vex.net/php/.

[6] Remote Control of Unix Netscape, Jamie Zawinski, URL - http://home.netscape.com/newsref/std/x-remote.html.

[7] MIME (Multipurpose Internet Mail Extensions) Part One: Mechanisms for Specifying and Describing the Format of Internet Message Bodies, N. Borenstein and N. Freed, RFC 1521, September 1993, URL - http://ds.internic.net/rfc/rfc1521.txt.

[8] Executing Shell Scripts Inside Mosaic, NCSA Mosaic Project, (1994), URL - http://www.ncsa.uiuc.edu/SDG/Software/Mosaic/Docs/executing-shell-scripts.html.