Harvard Journal of Law & Technology
Volume 25, Number 1 Fall 2011

OPEN SOURCE SEMICONDUCTOR CORE LICENSING

*Eli Greenbaum**

TABLE OF CONTENTS

I. INTRODUCTION

Semiconductor cores are not software in the conventional sense of that term. Even so, perhaps for lack of a readily available and more tailored framework, the tools used to license software have increasingly been applied to these works. This Article reviews the technology and structure of the semiconductor core industry and analyzes the application of open source license principles to semiconductor cores. Specifically, this Article focuses on the use of the GNU General Public License ("GPL"), arguably the most popular open source license.[1]

---

* Attorney, Yigal Arnon & Co., Jerusalem, Israel. J.D., Yale Law School; M.S., Columbia University. The author would like to thank Eyal Cohen-Melamed and Matanya Handler of Freescale Semiconductor for their help with the technical aspects of this Article, and Daniel Green for his helpful review and comments. Any mistakes or inaccuracies are, of course, the author's own.

1. *See* GLYN MOODY, REBEL CODE 301, 312–13 (2001) (stating that the "GPL is well established as the leading license in its field" and serves as the "de facto constitution for the entire free software movement"); *see also* Black Duck Software, *Open Source License Data*, OPEN SOURCE RESOURCE CENTER, http://osrc.blackducksoftware.com/data/licenses (last visited Dec. 21, 2011). Analysis of the Lesser General Public License ("LGPL") will also be made where appropriate. The text of the GPL version 3.0 is available at the Free Software Foundation website. *GNU General Public License*, GNU OPERATING SYSTEM, http://www.gnu.org/copyleft/gpl.html (last updated Sept. 20, 2011). The text of the LGPL version 3.0 is available at the same site. *GNU Lesser General Public License*, GNU OPERATING SYSTEM, http://www.gnu.org/licenses/lgpl.html (last updated Sept. 20, 2011). All references to the GPL and LGPL in this Article are to version 3.0 of each of these licenses, as these versions incorporate express terms that allow for licensing materials pursu-

As discussed below, the GPL is not perfectly suited to deal with the complexities of semiconductor cores. This Article shows where the provisions of the GPL clash with the technological and commercial needs of the semiconductor core industry, offers practical recommendations for using cores licensed pursuant to that framework, and concludes by offering suggestions for license provisions that would be more appropriate for the semiconductor core context. Analysis of issues raised by the GPL also highlights many of the more complex issues of open source licensing in the context of semiconductor cores, and can provide insights to help interpret and analyze problems raised by other licenses as well.

As the complexity of semiconductor chips grows, companies increasingly incorporate third-party intellectual property in the design of proprietary semiconductor devices.[2] These designs — colloquially referred to in the industry as "semiconductor intellectual property cores"[3] or simply "cores" — constitute discrete design units that can be incorporated in individual chip designs.[4] The licensing of semiconductor cores has become an essential part of the semiconductor industry, as the incorporation of ready-made design blocks can substantially reduce development time and costs.[5] Design core revenue was estimated at nearly $1.4 billion in 2007, up from $140 million in 1998 — thus growing at approximately twice the rate of the chip industry as a whole[6] — while estimates from 2008 have placed the value of the industry at between $1.4 billion and $2.0 billion.[7]

While most semiconductor cores are currently licensed on a proprietary basis, the growth of the industry has fostered the provision of cores under open source licenses. For example, OpenCores provides an online central repository for the development and distribution of open source cores.[8] Other commercial[9] and educational[10] entities also

---

ant to "semiconductor mask" laws. *See infra* text accompanying note 14. Of course, most of the issues described in this Article also apply to semiconductor cores licensed under older versions of the GPL.

2. *See generally* JEORGE S. HURTARTE ET AL., UNDERSTANDING FABLESS IC TECHNOLOGY 65–67 (2007); MICHAEL KEATING & PIERRE BRICAUD, REUSE METHODOLOGY MANUAL FOR SYSTEM-ON-A-CHIP DESIGNS 6–7 (3d ed. 2002) (describing the design challenges that mandate reuse of intellectual property).

3. *See, e.g.*, ILKKA TUOMI, INST. FOR PROSPECTIVE TECHNOLOGICAL STUDIES, EUROPEAN COMM'N JOINT RESEARCH CTR., THE FUTURE OF SEMICONDUCTOR INTELLECTUAL PROPERTY ARCHITECTURAL BLOCKS IN EUROPE 11 (Marc Bogdanowicz ed., 2009), *available at* http://cordis.europa.eu/fp7/ict/computing/includes/study.pdf [hereinafter IPTS STUDY].

4. Cores may also be referred to as "blocks," "macros," or simply "intellectual property." KEATING & BRICAUD, *supra* note 2, at 3.

5. *Id.* at 1.

6. CLAIR BROWN & GREG LINDEN, CHIPS AND CHANGE: HOW CRISIS RESHAPES THE SEMICONDUCTOR INDUSTRY 70 (2009).

7. *See* ITPS STUDY, *supra* note 3, at 15 tbl.1.

8. OPENCORES, http://opencores.org (last visited Dec. 21, 2011). OpenCores generally recommends that users employ the LGPL or Berkeley Software Distribution ("BSD") li-

provide core designs licensed under commonly used open source licenses. However, the incorporation of open source cores in commercial products has remained limited, mostly due to concerns regarding dependability, maintenance, and support.[11] Of course, it follows that these practical concerns would be less pressing to the extent that commercial entities maintain and support open source cores. The commercial industry may be able to provide licensees with open source cores that have been tested and validated, especially with regard to more standardized functionalities.[12] As the industry grows and matures, the legal issues raised by such open source licensing issues will likely achieve more prominence.

The application of open source software licenses (such as the GPL) to semiconductor cores may at first impression seem attractive, since the early stages of semiconductor device design bear strong resemblance to software programs.[13] Additionally, the text of the GPL

censes. *See What License Is Used for OpenCores?*, *Frequently Asked Questions*, OPENCORES, http://opencores.org/opencores,faq#whatlicense (last visited Dec. 21, 2011). Many projects available on OpenCores also make use of the GPL license, as can be seen by looking at the licensing option on the projects page. *See Projects*, OPENCORES, http://opencores.org/projects (last visited Dec. 21, 2011). For a comparison of the functionality of several open source cores with various commercial cores, see Tong et al., *Soft-Core Processors for Embedded Systems*, 18TH INT'L CONF. ON MICROELECTRONICS 170 (2006).

9. In March 2006, Sun Microsystems made the code for the entire UltraSPARC T1 processor available under version 2.0 of the GPL. *See About OpenSPARC*, OPENSPARC, http://www.opensparc.net/about.html (last visited Dec. 21, 2011). Another core available under an open source model is the LEON 32-bit microprocessor originally developed by the European Space Agency, and subsequently by Aeroflex Gaisler AB. Aeroflex Gaisler now offers a library of cores under a dual licensing model, where the core may be obtained pursuant to the GPL or a commercial license. *See generally IP Cores*, AEROFLEX GAISLER, http://www.gaisler.com/cms/index.php?option=com_content&task=section&id=5&Itemid=51 (last visited Dec. 21, 2011).

10. SimpleCores provides cores for educational purposes. *See* SIMPLECORES, http://www.simplecores.co.cc (last visited Dec. 21, 2011). The cores are generally licensed under the Creative Commons Attribution-NonCommercial-ShareAlike 3.0 license. *Terms and Licenses*, SIMPLECORES, http://www.simplecores.co.cc/terms-and-licenses (last visited Dec. 21, 2011). The University of Toronto makes available a soft core processor under the terms of the GPL. UT NIOS, http://www.eecg.toronto.edu/~plavec/utnios.html (last visited Dec. 21, 2011). Additionally, the Pierre et Marie Curie University provides libraries for VLSI design under the GPL. *See Alliance*, LABORATOIRE D'INFORMATIQUE DE PARIS 6, http://www-soc.lip6.fr/recherche/cian/alliance (last visited Dec. 21, 2011).

11. *See* Declan Staunton, *Successful Use of an Open Source Processor in a Commercial ASIC*, DESIGN & REUSE, http://www.design-reuse.com/articles/12145/successful-use-of-an-open-source-processor-in-a-commercial-asic.html (last visited Dec. 21, 2011) (stating that "[i]mmaturity of designs, lack of support, licensing and warranty concerns would normally ensure open source IP cores" are not used in commercial ASIC design). For the importance of maintenance and support to customers of semiconductor IP, see IPTS STUDY, *supra* note 3, at 47 ("IP users need reliable and predictable vendors, who can provide support and maintain their IP as long as the user needs it. Such product lifecycle management typically requires considerable resources.").

12. IPTS STUDY, *supra* note 3, at 52 (stating that open source development could become "highly important" if "standardized interfaces and a relatively stable 'core'" could be developed).

13. *See infra* text accompanying notes 21–22.

states that it can be applied to the license of materials pursuant to "copyright-like laws that apply to . . . semiconductor masks."[14] However, as detailed below, the life cycle of the semiconductor core involves several other kinds of expression in addition to the early software-program-like stages. Each of these stages demands a different analysis under copyright law, as well as under other intellectual property regimes. Thus the interpretation of the GPL in the context of semiconductor core licensing is not always straightforward.[15]

Part II provides a brief technical survey of the standard implementation of a semiconductor core in a physical device. As that Part shows, the semiconductor core begins its complex life as a software model, and only slowly matures into hardware. Thus the use of an open source license can raise legal questions far before the distribution of an actual semiconductor device. Part III provides a brief summary of some relevant sections of the GPL and applicable case law. Parts IV–VI apply several provisions of the GPL to the licensing of semiconductor cores. These Parts show that the provisions of the GPL are not always appropriate in the semiconductor industry setting. Part VII concludes and suggests how open source licenses may be tailored to address the problems raised by semiconductor cores.

II. TECHNICAL BACKGROUND

A semiconductor core is the design of a discrete functional unit of an integrated circuit.[16] In other words, rather than design entire chips, licensors of semiconductor cores produce modular designs that can be incorporated into larger chip models. Examples of semiconductor cores include designs for microprocessors, memory components, or blocks to implement USB or Ethernet connectivity.[17] These discrete

14. *See GNU General Public License*, *supra* note 1, § 0. To some extent, other definitions in the GPL have also been drafted in a sufficiently abstract manner such that they can be interpreted and applied coherently in the semiconductor design context. For example, § 1 defines source code as "the preferred form of the work for making modifications to it," and defines object code as "any non-source form of a work." Additionally, § 0 defines "Program" as "any copyrightable work licensed under this License." The high-level abstraction of these definitions enables their application in the context of semiconductor core licensing. The application of other sections of the GPL to semiconductor cores, however, may require interpretation of provisions tailored specifically to the standard software context. *See, e.g.*, *infra* text accompanying note 114 (applying the "System Library" exception in the semiconductor core context).

15. The available literature on the application of open source licenses to semiconductor cores may be generously described as scant. John Ackermann discusses the philosophy and goals of open source hardware, but focuses on recommending the TAPR Open Source Hardware License. John R. Ackermann, *Toward Open Source Hardware*, 34 U. DAYTON L. REV. 183, 183–85 (2009). In the course of that discussion, Ackermann briefly touches on the application of the GPL to hardware designs. *Id.* at 192–93.

16. *See* KEATING & BRICAUD, *supra* note 2, at 3.

17. *Id.*; *see also* IBM, ASIC DESIGN METHODOLOGY PRIMER 1 (1998), http://web.cecs.pdx.edu/~greenwd/asic_primer1.pdf [hereinafter IBM PRIMER].

design units are incorporated by the licensee into the design of a larger hardware device.

In the mid-1980s, rising fabrication costs, the establishment of independent contract manufacturers, and the emergence of the fabless-foundry business model encouraged the growth of semiconductor core licensing.[18] Microchip companies continued to design chips, but started to rely on foundries to focus on the complex technology of device manufacturing.[19] Freed from the burdens and costs of manufacturing, design firms were able to turn to the even more specialized business of producing discrete units that could be incorporated into larger chip designs.[20] The business model of semiconductor core licensing, which is premised on the reusability and recycling of designs, has promised to increase the speed and efficiency of semiconductor chip production.[21]

The design of a semiconductor core is expressed in a wide variety of formats in the production process. The successful core design starts as a software-like program, but is then transformed into machine-readable form and finally into a graphic blueprint format. The core design's first stage is in a hardware description language ("HDL"), which is a software-like representation of the functionality of an actual hardware device.[22] HDLs can resemble software to such a degree that attempts have been made to replace HDLs with standard software languages.[23] HDL code describes the logic of the device, but does not detail the electronic components of the device that implement this logic (which is the function of the "netlist" described in the next paragraph). In fact, the abstract logic described by HDL code can often be implemented by a broad range of circuits.[24] The specific circuit that will be used to implement the HDL code will only later be chosen by design software according to criteria and constraints imposed by the designer, such as the desired power, size, and performance of the de-

18. *See* BROWN & LINDEN, *supra* note 6, at 39; JIM TURLEY, THE ESSENTIAL GUIDE TO SEMICONDUCTORS 96–101 (2003) (providing short introductions to the history and economics of the fabless-foundry model).

19. TURLEY, *supra* note 18, at 115.

20. *See* HURTARTE, *supra* note 2, at 47, 106–07, 109. According to Hurtarte, "the emergence of the fabless semiconductor industry, including a full ecosystem of suppliers, enabled all of the required elements to design and manufacture a chip to be sourced from third parties: electronic design automation (EDA) tools, cell libraries and IP blocks on the design side; and package, assembly and test services and wafer fabrication on the manufacturing side." *Id.* at 47.

21. *See generally* BROWN & LINDEN, *supra* note 6, at 69.

22. For a comparison between HDL and software programming languages, see TURLEY, *supra* note 18, at 41–44. Turley also discusses recent attempts to replace HDL descriptions with widely-used software languages, such as C programming. *Id.* at 44.

23. *Id.* at 41–44.

24. IBM PRIMER, *supra* note 17, at 7 (stating that there are "potentially hundreds, or even thousands, of different combinations of logic circuits that can implement the same logical function").

vice.[25] In summary, the HDL description of a core can be compared to the source code of a software program: both describe logical processes in a manner that can be understood, copied, and modified by humans but that to some extent obscures the actual steps necessary for machines to implement this logic.

The next stage of chip design is termed "logic synthesis."[26] In this stage, HDL code is converted by electronic design automation ("EDA") software into a "netlist," which is a list of the electronic components that would make up the functionality described by the HDL code.[27] In other words, this stage creates a description of the electronic circuit, including all individual components, which will eventually be implemented in the physical device. In producing the netlist, the HDL code is combined with a "synthesis library" that specifies the characteristics of individual electronic elements.[28] This library is typically proprietary to a specific foundry and its particular fabrication process.[29] The process of creating the netlist is akin to compiling software source code: both take an abstract description of a logical process and transform it into a form that can be implemented using specified technology.[30] The creation of the netlist allows for the

25. *Id.* Devices that aim to minimize power consumption, for example, will implement HDL code differently than devices that aim to maximize performance. *See* JUAN-ANTONIO CARBALLO, CHIP DESIGN FOR NON-DESIGNERS: AN INTRODUCTION 59 (2008) (describing the potential trade-off between power consumption and speed with regard to selection of library gates).

26. CARBALLO, *supra* note 25, at 57.

27. *See* TURLEY, *supra* note 18, at 45 (defining a netlist as "a tangled list of which electrical circuits are connected to which other circuits"); *id.* at 36 (defining EDA as "the business of designing and selling chip-design software"); IBM PRIMER, *supra* note 17, at 7 ("The output of the synthesis process is a list of circuit instances interconnected in a manner that implements the logical function of the design. This list of interconnected circuit instances is called a netlist . . . .").

28. IBM PRIMER, *supra* note 17, at 7; *see also* RAKESH KUMAR, FABLESS SEMICONDUCTOR IMPLEMENTATION 145 (2008) (providing a description of the detail included in libraries).

29. *See* HURTARTE, *supra* note 2, at 72 ("Foundry-provided SIP products are usually not technically portable, or their use in another foundry process may be restricted by license."). Cadence Design Systems, Inc., for example, offers libraries for the manufacturing of devices at Taiwan Semiconductor Manufacturing Company, Ltd. ("TSMC"), one of the world's largest foundries. Use of the libraries is subject to license agreements, copies of which are available online. *See Cadence and TSMC Library Distribution*, CADENCE DESIGN SYS., INC., http://www.cadence.com/Alliances/ip_program/Pages/tsmc_lib.aspx (last visited Dec. 21, 2011). Section 1.1 of the Front-End Library Usage Agreement, for example, provides that "fabrication of the products containing the Library shall be exclusively at" TSMC. *Library Usage Agreement*, CADENCE DESIGN SYS., INC., http://www.cadence.com/Alliances/ip_program/Documents/TSMC_Library_License_front-end.pdf (last visited Dec. 21, 2011). Standard cell libraries have also been made available under open source licenses. *See, e.g.*, Graham Petley, *VLSI and ASIC Technology Standard Cell Library Design*, VLSI TECH., http://www.vlsitechnology.org (last updated Sept. 22, 2008) (providing standard cell libraries pursuant to the GPL).

30. For a description of how compilers for software programs implement high-level programming languages in a machine code that is specific to a defined technology, see

simulation and verification of the actual circuit,[31] as well as for the creation of the blueprints for a physical device.[32] The netlist, however, is not readily understandable or modifiable by humans.[33] Moreover, the netlist contains only an idealized description of electronic components and, as such, does not itself contain sufficient information to actually enable the manufacture of a semiconductor chip.[34]

The next group of manufacturing stages is termed "physical design," and these steps translate the design into a manufacturable blueprint.[35] The physical design stages, of which there are several, allow for the manufacturing of an actual semiconductor device according to the requirements and constraints of a specific manufacturing process. In the "floor planning" stage, groups of components are assigned to different areas of the device, and this allows the designer to consider the impact of the physical location of these components.[36] The "place and route" stage then involves the placement of all components on the device and the connection of these components.[37] Completed device designs often require several iterations of this process.[38] Physical design also involves mapping the power and timing of the chip.[39] The physical design stages are often outsourced to a third-party company with the required tools and expertise to execute this process efficiently.[40] The entire design process typically results in a "GDSII file," a graphic format that specifies the complete layout of the final chip design.[41] The GDSII file is provided to a foundry for the actual physical fabrication of the chip.[42]

CHARLES PETZOLD, CODE: THE HIDDEN LANGUAGE OF COMPUTER HARDWARE AND SOFTWARE 353 (2000).

31. *See* CARBALLO, *supra* note 25, at 59 (stating that "[m]odern design tools allow the performance of myriad operations on a netlist, to verify a large number of aspects of a design").

32. *See* TURLEY, *supra* note 18, at 45 (describing the operation of a floor planning program on a netlist to produce an optimal arrangement for all the parts of a chip).

33. *Id.* at 45 ("A netlist is generally only readable by a computer; it's too convoluted and condensed to be of any use to a person.").

34. *See id.* at 45 ("The netlist is just an intermediate stop along the way to a new chip."); Ackermann, *supra* note 15, at 189–90 (describing the circuit board layout process that must follow the creation of a netlist).

35. BROWN & LINDEN, *supra* note 6, at 64 ("The final stage, physical design, involves the translation of the abstract version into a map of actual wires and devices interconnecting across multiple layers on the silicon surface"); Ackermann, *supra* note 15, at 189.

36. IBM PRIMER, *supra* note 17, at 14, 19.

37. TURLEY, *supra* note 18, at 46; IBM PRIMER, *supra* note 17, at 19. The "place and route" stage may also be referred to as the "layout" stage. *See id.*

38. *See* TURLEY, *supra* note 18, at 48 (discussing the difficulties of completing place and route in a single iteration).

39. *See* CARBALLO, *supra* note 25, at 142–44.

40. HURTARTE, *supra* note 2, at 50, 63–64 (describing the economics of separating the initial design phases from the physical design); s*ee also* KUMAR, *supra* note 28, at 82.

41. TURLEY, *supra* note 18, at 54.

42. For convenience, this Article has focused on the process for manufacturing Application Specific Integrated Circuits ("ASICs"), though similar questions may arise in the context of Field Programmable Gate Arrays ("FPGAs"). A FPGA is a general-purpose chip

Cores may be licensed at different stages of the design process described above. A "soft core" is generally provided in a high-level HDL language, but may also be provided in netlist format.[43] In other words, a soft core is essentially a software-like representation of the functionality of a hardware design, rather than the actual device schematics implementing the design. Soft cores allow for customization, as the software-like representation can often be modified and tuned by the licensee.[44] As soft cores are not tailored to a specific manufacturing process, they can generally be used to manufacture devices in a variety of different foundries.[45] At the same time, soft cores have certain disadvantages. First, the use of soft cores requires an investment of time and expertise.[46] Second, since soft cores are not optimized for a specific manufacturing process, quantification of their performance is often difficult.[47] Examples of soft cores generally include designs that will be implemented across a variety of manufacturing technologies. These may include processor cores, which are implemented by a range of companies in a variety of production processes.[48]

In contrast to soft cores, "hard cores" are generally provided in the GDSII format.[49] A hard core, in other words, resembles a blueprint of an actual semiconductor device. Hard cores theoretically can be fitted into the design of the entire device like a puzzle piece, and thus may provide for the lowest risk when integrating the core into the device.[50] Hard cores are usually optimized for a specific fabrication process and are generally not portable across different foundries; they can neither be modified nor customized.[51] Examples of cores that are typically provided in hard format include designs that are sensitive to changes in configuration or manufacturing process, such as cores that provide analog functionality or memory.[52]

In summary, semiconductor cores represent the functionality of a hardware device, but that functionality can be expressed in a variety of formats. During the production of the typical semiconductor device, cores are taken through a correspondingly wide range of expression — from the first software-like stages, through the intermediate

---

whose hardware can be reconfigured by users to create many different chip designs. *See generally* TURLEY, *supra* note 18, at 160–61.

43. HURTARTE, *supra* note 2, at 67.

44. *Id.* at 109. For example, the soft core can be modified to enable or disable functions in the design, depending on the customer's needs. *Id.*

45. *Id.* at 67.

46. *Id.* at 109.

47. *See id.* at 67–68.

48. *See id.* at 69.

49. *Id.* at 68.

50. *Id.* at 110.

51. *Id.* at 68, 110.

52. *See id.* at 110, 113–15; *see also* KEATING & BRICAUD, *supra* note 2, at 25–26.

netlist representation, and culminating in the graphic formats that can be used to manufacture devices.

III. THE GPL AND LGPL

Version 1.0 of the GPL was introduced by Richard Stallman in 1989; since then the GPL has arguably become the most widely used open source license.[53] Prominent software licensed under the GPL includes the Linux kernel[54] and the GNU Compiler Collection ("GCC").[55] The GPL is published and maintained by the Free Software Foundation, which also promotes the use and development of open source software.[56]

The GPL grants users broad rights to freely copy, modify, and distribute licensed programs, but also imposes certain requirements and conditions.[57] The distributor of a GPL-licensed work, for example, must make the source code of that work available under the terms of the GPL.[58] In other words, recipients of the work are granted the right to copy, modify, and distribute the source code of the work, and those rights cannot be restricted by the licensor.[59] One of the distinguishing features of the GPL is its "copyleft" provisions, which generally require that all works "based on" a licensed work also be released under the same license terms.[60] The scope and application of the GPL copyleft provisions — how broadly should the term "based on" be interpreted, for example — is far from settled and remains a contentious topic.[61] The uncertainty surrounding the scope of these provisions has meant that commercial software distributors have been

53. MOODY, *supra* note 1, at 26–28, 301; *Open Source License Data*, *supra* note 1.

54. *GNU General Public License*, LINUX KERNEL, http://git.kernel.org/?p=linux/kernel/git/torvalds/linux.git;a=blob_plain;f=COPYING (last visited Dec. 21, 2011).

55. RICHARD M. STALLMAN & THE GCC DEVELOPER COMMUNITY, GCC, THE GNU COMPILER COLLECTION 647–56 (2011), http://gcc.gnu.org/onlinedocs/gcc-4.6.1/gcc.pdf.

56. John Sullivan, *Free Software Is a Matter of Liberty, Not Price*, FREE SOFTWARE FOUND., http://www.fsf.org/about (last visited Dec. 21, 2011).

57. *See generally*, *GNU General Public License*, *supra* note 1.

58. *Id.* §§ 4–6.

59. *Id.* (setting forth the conditions for the redistribution of a licensed work, including in modified or non-source form).

60. *Id.* Section 5 of the GPL allows licensees to convey a "work based on the Program . . . in the form of source code," subject to certain conditions, so long as the source code is provided pursuant to the licensing terms of the GPL. *Id.* § 5. Section 6 of the GPL allows licensees to convey a "covered work" (defined as the licensed work or a work "based on" the licensed work) in object code, provided that certain source code of the covered work is also distributed pursuant to the terms of the GPL. *Id.* § 6.

61. See *infra* notes 86–88 for a brief summary of positions that commentators have taken on the topic; *see also* Malcolm Bain, *Software Interactions and the GNU General Public License*, 2 INT'L FREE AND OPEN SOURCE SOFTWARE L. REV. 165 (2010), *available at* http://www.ifosslr.org/ifosslr/article/view/44/74 ("The so-called 'GPL linking' debate has been raging for the last 18 years, and probably will go on for . . . quite a few more.").

wary of incorporating GPL-licensed software in their proprietary products.[62]

Case law interpreting the GPL in the United States has been limited.[63] In *Progress Software Corp. v. MySQL AB*,[64] a case that implicated the scope of the copyleft provision, MySQL sued Progress Software for distributing a database product that linked to MySQL's GPL-licensed code without also distributing the source code of that product.[65] MySQL requested a preliminary injunction enjoining Progress Software from distributing MySQL's software.[66] The court declined to grant the preliminary injunction.[67] Despite the fact that the court seemed to accept the enforceability of the GPL's copyleft provisions, the court stated there remained a factual dispute as to whether Progress Software's product constituted "a derivative or an independent and separate work"[68] under the GPL — in other words, whether Progress Software's product was "based on" MySQL's GPL-licensed product and therefore subject to the copyleft provisions of the GPL. No other court decisions in the United States have yet interpreted the scope of the GPL's copyleft provisions, though other decisions have involved the effect and enforceability of the GPL,[69] including in the context of trademark infringement[70] and antitrust law.[71]

The Free Software Foundation also publishes and maintains the Lesser General Public License ("LGPL"), a license that contains more permissive copyleft provisions than the GPL. As with the GPL, the LGPL also generally requires that works based on an LGPL-licensed

---

62. *See* Lawrence Rosen, *The Unreasonable Fear of Infection*, ROSENLAW & EINSCHLAG (2001) http://rosenlaw.com/html/GPL.PDF; *see also* HEATHER J. MEEKER, THE OPEN SOURCE ALTERNATIVE: UNDERSTANDING RISKS AND LEVERAGING OPPORTUNITIES 183–84 (2008) (discussing the consequences of the reach of the GPL copyleft provisions on software providers).

63. *See generally* THE INTERNATIONAL FREE AND OPEN SOURCE SOFTWARE LAW BOOK (2011), http://ifosslawbook.org (discussing case law in other countries regarding the enforcement and interpretation of the GPL). Germany especially has produced relatively extensive case law regarding the GPL. *See id.*

64. 195 F. Supp. 2d 328 (D. Mass. 2002); s*ee also* Affidavit of Eben Moglen, Progress Software Corp. v. MySQL AB, No. 01-CV-11031 (D. Mass. Feb. 26, 2002), *available at* http://www.gnu.org/press/mysql-affidavit.pdf (summarizing some of the facts of the case).

65. Affidavit of Eben Moglen, *supra* note 64.

66. *Progress Software Corp.*, 195 F. Supp. 2d at 329.

67. *Id.* at 329–30.

68. *Id.* at 329.

69. *See* Software Freedom Conservancy v. Best Buy, 783 F. Supp. 2d 648, 652 (S.D.N.Y. 2011) (default judgment awarding treble statutory damages for the distribution of BusyBox software in violation of the GPL).

70. *See* Planetary Motion, Inc. v. Techsplosion, Inc., 261 F.3d 1188, 1198 (11th Cir. 2001) ("Software distributed pursuant to [the GPL] . . . is not necessarily ceded to the public domain and the licensor purports to retain ownership rights, which may or may not include rights to a mark.").

71. *See* Wallace v. Free Software Found., Inc., 1:05-CV-0618-JDT-TAB, 2006 WL 2038644 (S.D. Ind. Mar. 20, 2006).

program be relicensed under the same terms.[72] At the same time, and subject to certain conditions, the LGPL permits proprietary applications to be combined and distributed with LGPL-licensed libraries, and does not require that the source code of the proprietary application be disclosed.[73] As the LGPL is more permissive than the GPL, the Free Software Foundation has not always encouraged the use of the LGPL.[74]

The remainder of this Article will apply the provisions of the GPL in the context of semiconductor cores, showing how the technological process described in Part II complicates the application and analysis of the GPL. The GPL provides that its obligations are triggered by the distribution of a covered work; Part III examines this trigger in the context of semiconductor industry practice. Part IV analyzes the copyleft provisions of the GPL as applied to semiconductor cores. Part V advances a counterintuitive argument that the GPL may be read as extending to distributions of physical devices that include GPL-licensed material.

IV. DISTRIBUTION

Rights and obligations under the GPL often hinge on whether a work has been "conveyed." Section 6 of the GPL, for example, permits licensees to convey the object code of a licensed work, provided that the corresponding source code of the work is also delivered.[75] In other words, it is the act of "conveying" a work that triggers the requirement to provide the source code. The interpretation of the term "convey" is not settled. For example, the Free Software Foundation

72. Section 2 of the LGPL generally provides that users may modify and convey a licensed work if the modified work is also distributed under the LGPL (or, alternatively, under the more restrictive conditions of the GPL). *GNU Lesser General Public License*, *supra* note 1.

73. *See id.* § 4(d) (requiring only disclosure of the "Minimal Corresponding Source," which is generally defined as the source code of the Library but not the source code of an application linked to or combined with the Library).

74. *See Why You Shouldn't Use the Lesser GPL for Your Next Library*, FREE SOFTWARE FOUND., http://www.gnu.org/philosophy/why-not-lgpl.html (last updated Sept. 20, 2011).

75. *See GNU General Public License*, *supra* note 1, § 4 (regarding the conveying of unmodified versions of a licensed work); *id.* § 5 (regarding conveying modified source code); *id.* § 10 (providing that "[e]ach time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License"); *see also* MEEKER, *supra* note 62, at 233 (stating that distribution is what triggers the requirements of the GPL). The term "convey" in the GPL replaces the term "distribute" used in previous versions of the license. The motivation of this change was to replace language grounded in United States law with a more factual description of behavior that triggers GPL obligations. *See Opinion of the Denationalization of Terminology*, FREE SOFTWARE FOUND., http://gplv3.fsf.org/denationalization-dd2.html (Aug 3, 2006, 4:04 PM) [hereinafter FSF, *Denationalization*]. In this Article, given the design process described above, "source code" should generally be taken to mean the HDL code of the core, though the term has been defined broadly enough to include schematics, mechanical drawings, and other documentation as well. *See supra* note 14.

has asserted that distribution to an off-site independent contractor constitutes distribution,[76] while other commentators have proposed a more narrow understanding of the term.[77] While the language of the GPL itself permits users to "modify a private copy" without triggering copyleft rights and obligations, what constitutes a "private copy" is likely to vary across jurisdictions.[78]

Semiconductor intellectual property cores are almost certain to be conveyed to third parties during the design and fabrication process. As detailed in Part II, the typical lifecycle of the core involves the distribution of designs to third parties in either soft or hard format. Fabless companies increasingly outsource physical design stages to independent contractors.[79] Certain other engineering stages, such as testing and characterization, may be outsourced as well.[80] In addition, fabless companies are almost certain to provide the completed GDSII layout to a foundry for fabrication. Even fabless companies that eschew including GPL cores in their commercial products may find themselves

---

76. *See Frequently Asked Questions about the GNU Licenses*, GNU OPERATING SYS., http://www.gnu.org/licenses/gpl-faq.html (last updated Dec. 12, 2011, 7:07 PM) [hereinafter *GNU, FAQ*]. In order to provide clarity on the issue, § 2 of version 3.0 of the GPL added an exception pursuant to which conveyance to an outside contractor does not grant the contractor rights under the GPL. This exception provides that "[y]ou may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, *provided that you comply with the terms of this License in conveying all material for which you do not control copyright*." *GNU General Public License*, *supra* note 1, § 2 (emphasis added). Unfortunately, this exception will likely not be helpful for the typical fabless company using third party cores, since only works to which a user controls copyright may be provided to an outside contractor not pursuant to the GPL. A semiconductor design will likely incorporate third party technology libraries and may contain other third party cores to which a fabless company would not control copyright, and that company would not have the right to license these components pursuant to the GPL. *See infra* Part V. Moreover, the exception requires that any such outside contractors operate under the "direction and control" of the company. This requirement is likely to prevent application of this exception in most situations, since the high standard of "direction and control" is often the hallmark of an employment relationship rather than of independent contractor status. *See, e.g.*, Rev. Rul. 87-41, 1987-1 C.B. 296 (Internal Revenue Serv. 1987) (stating that under common law rules "generally the relationship of employer and employee exists when the person or persons for whom the services are performed have the right to *control and direct* the individual who performs the services" (emphasis added)).

77. *Comments of the New York City Bar Association to GPL Version 3*, 61 THE REC. OF THE ASS'N OF THE BAR OF THE CITY OF N.Y. 325, 329 (2006) (stating that the question should be "whether the modified work is being used solely for the benefit of the licensee, regardless of whether the use is by a contractor or an employee"); *see* MEEKER, *supra* note 62, at 233–36.

78. The GPL's definition of "convey" incorporates an express reference to applicable copyright law: the term "propagate" is generally defined as an act that "applicable copyright law" would deem an infringement. *GNU General Public License*, *supra* note 1 § 0; *see also* FSF, *Denationalization*, *supra* note 75 (stating that the GPL leaves "the line drawn between licensees and other parties for determination under local law").

79. *See* HURTARTE, *supra* note 2, at 16–17 (discussing outsourcing in the context of fabless semiconductor firms); *id.* at 49–51 (discussing the outsourcing of physical design); KUMAR, *supra* note 28, at 82, 93–95 (discussing which stages of design and development may be outsourced).

80. *See* HURTARTE, *supra* note 2, at 27; KUMAR, *supra* note 28, at 82.

inadvertently distributing open source materials. For example, a core may be available under a dual-license model, where it can be used pursuant to an open source license, or under a proprietary license for commercial purposes.[81] A fabless company may initially use such a core for evaluation purposes under the open source license, intending to pay for the commercial license only later in the development process. The company could work through several design stages with the open source core, purchasing the commercial license only when it has ensured that the core can be efficiently incorporated into its product. Conveyance of the open source core used during evaluation and development stages would trigger the provisions of the GPL.

In the absence of judicial guidance, it may be useful to postulate a spectrum of acts, ranging from those that should not involve "conveying" to those actions that may. For example, the outsourcing of physical design should not be considered "conveyance" because the contractor is likely to assign all of its intellectual property rights in the design to the company, and is also likely to be subject to strict confidentiality obligations.[82] These requirements should indicate that work performed by the outside contractor is only the modification of a "private copy." Further down the spectrum, some development or fabrication contracts are likely to contain more complex intellectual property and confidentiality provisions. The developer or foundry may insist on ownership of developments related to its own technology as well as compliance with related confidentiality provisions.[83] In this situation, the developer or foundry would no longer be acting solely for the benefit of the fabless company, and perhaps this would be a "convey-

81. For example, Aeroflex Gaisler provides an "evaluation version" of semiconductor cores under the GPL but states that commercial licenses must be purchased for applications where the conditions of the GPL are not appropriate. *See supra* note 9. For a short discussion of dual-license commercial models, see LAWRENCE ROSEN, OPEN SOURCE LICENSING: SOFTWARE FREEDOM AND INTELLECTUAL PROPERTY LAW 262 (2005). Commercial entities using a dual-licensing model would probably use the GPL (or another license with strong copyleft provisions) as an open source license, since using more permissive licenses would undermine the viability of the business model. *See* Michael Olson, *Dual Licensing*, *in* OPEN SOURCES 2.0: THE CONTINUING EVOLUTION 83 (Chris Dibona et al. eds., 2005).

82. *See* HURTARTE, *supra* note 2, at 190 (stating that fabless companies should "relentlessly" use non-disclosure agreements and that consultant agreements with designers should include intellectual property assignments to the fabless companies).

83. For example, § 5.3 of the standard terms and conditions for foundry services from United Microelectronic Corporation ("UMC"), one of the world's largest foundries, provides that neither party is restricted from using "improvements . . . provided, derived and/or developed in whole or in part by or on behalf of such party." *UMC Wafer Foundry Standard Terms and Conditions*, UNITED MICROELECTRONICS CORP., http://www.umc.com/english/pdf/umctermsDec2010.pdf (last visited Dec. 21, 2011). The effect of this provision is that some intellectual property developed during the provision of services may be the property of the foundry rather than of the fabless company. Section 5.2 of these terms and conditions, while generally providing that UMC will treat information provided by purchasers of foundry services as confidential, also allows UMC to provide its own vendors with certain anonymized confidential information generated by those foundry manufacturing processes. *Id.*

ance" pursuant to the GPL. Finally, providing the core to a customer typifies a commercial vendor-customer relationship and, as such, is likely to trigger the rights and obligations of the GPL. It should also be noted that, as discussed in greater detail in Part VI, it is not impossible to eliminate the risk that the provision of the completed physical device will also be seen as involving the "conveyance" of an embodied core.

As discussed above, the typical fabless company will invariably find that it needs to distribute the design of its device to third parties — whether for the purpose of design, fabrication, or testing. In the absence of guiding law, it is not currently possible to provide definitive advice as to whether any of these actions constitutes conveyance under the terms of the GPL. In light of such uncertainty, companies should consider the potential risk involved in these activities. Companies can increase the probability that any conveyance will be seen as the distribution of a "private copy" by incorporating strict intellectual property and confidentiality provisions in any outsourcing or contractor agreements.

V. LINKING

This Part analyzes the application of the copyleft provisions of the GPL in the context of semiconductor cores. According to these provisions, a distributor of a GPL-licensed work must provide the source code of the work under GPL terms, as well as the source code of all works that are "based on" the original work.[84] These provisions, according to the Free Software Foundation, protect the freedom to use, modify, and redistribute licensed works.[85]

While the practical application of these provisions has been the subject of extensive commentary, some generally accepted contours of analysis have emerged. First, most commentators believe that the GPL copyleft obligations should be limited to works that are derivative works of the original licensed works.[86] Second, the GPL itself

84. *GNU General Public License*, *supra* note 1.

85. *See What Is Copyleft?*, GNU OPERATING SYS., http://www.gnu.org/copyleft/copyleft.html (last updated Sept. 20, 2011).

86. *See* ROSEN, *supra* note 81, at 120; Lothar Determann, *Dangerous Liaisons — Software Combinations as Derivative Works?: Distribution, Installation, and Execution of Linked Programs Under Copyright Law, Commercial Licenses, and the GPL*, 21 BERKELEY TECH. L.J. 1421, 1491 (2006) (stating that "the context of the GPL favors an interpretation of the term 'derived work' to mean 'a derivative work as defined by the Copyright Act'"); Mikko Välimäki, *GNU General Public License and the Distribution of Derivative Works*, J. INFO. L. & TECH. (2005), http://www2.warwick.ac.uk/fac/soc/law/elj/jilt/2005_1/valimaki (stating that "what is meant by a derivative work [in the GPL] is in the end defined by the interpretation of copyright law"). The Free Software Foundation has stated that it considers the phrase "works based on the Program" in the GPL to be similar though perhaps not identical to the definition of derivative work under copyright law. *See* FSF, *Denationalization*, *supra* note 75. Some commentators believe that the practical positions taken by the Free

provides that the copyleft provisions should generally not apply to the compilation of a GPL-licensed work together with "separate and independent works."[87] Unfortunately, distinguishing derivative works from compilations under copyright law is not always easy.[88] As such, the implementation of these principles has been the subject of fierce debate.[89]

In analyzing the application of these provisions, and in the interest of simplicity, this Article will take a derivative work to generally mean a work which substantially copies from a pre-existing work, but which also contains significant changes to that pre-existing material.[90] Using that general principle, this Article will analyze three situations where a GPL-licensed core is incorporated into a device design. As with software, the interpretation of these provisions in the distinctive context of semiconductor core licensing will undoubtedly involve much debate, both regarding the appropriate legal rules as well as with respect to their application in complex technical situations. It is not the intention of this Article to attempt to thoroughly apply the GPL copyleft provisions in an entirely new technological area; rather, this Article will provide a general framework of analysis that can be further developed in more specific circumstances.

A. Hard Cores

This Part analyzes the integration of a proprietary device design with a GPL-licensed hard core. As described above, a hard core is the blueprint of a semiconductor device layout in graphic format. A relatively simple example of the integration of a hard core into a proprietary design would be the situation of a fabless company that designs a chip for cellular phones, and adds a GPL-licensed "input/output"

Software Foundation with regard to the implementation of this view are inaccurate. *See, e.g.*, Determann, *supra*, at 1488 (stating that "the drafters of the GPL intended to cover software combinations that would not qualify as derivative works"); Välimäki, *supra*, § 3.4 (describing the position of the Free Software Foundation as "remote to both copyright law and the GPL").

87. *GNU General Public License*, *supra* note 1, § 5 (providing that "[i]nclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate"); *see also GNU, FAQ*, *supra* note 76 (stating that the GPL permits a number of "separate programs" to be distributed together).

88. *See* 1 MELVILLE B. NIMMER & DAVID NIMMER, NIMMER ON COPYRIGHT § 3.08 (2011) [hereinafter NIMMER ON COPYRIGHT] (describing the difficulty of distinguishing derivative works from collective works).

89. *See* Bain, *supra* note 61, at 165.

90. *See* NIMMER ON COPYRIGHT, *supra* note 88, § 3.01 (stating that a "work is not derivative unless it has *substantially* copied from a prior work"); *id.* § 3.02 (stating that, unlike collective works, a "derivative work involves recasting or transformation, *i.e.*, changes in the pre-existing material"); Determann, *supra* note 86, at 1437 (stating that software combinations "qualify as derivative works only if they are sufficiently permanent, contain significant amounts of existing copyrighted works and involve significant and creative changes to such pre-existing works").

("I/O") core in hard format. I/O cores allow connections to and from external devices such as phone keypads, typically through a standard interface.[91]

In this example, the fabless company receives the GPL-licensed I/O hard core as a separate file that describes a three-dimensional layout of the component.[92] The layout of the hard core could simply be fixed in an area of the design left vacant to accommodate the functionality of the core. Integration of the hard I/O core would require specified interfaces, but would not require material changes to the internal design of the remainder of the cellular chip. The fabless company would then distribute the whole device design — including the GPL-licensed core — to a third party, such as a foundry or an independent contractor.

In this (relatively) simple example, the rights and obligations of the GPL would likely be restricted to the hard core, and should not extend to the remainder of the design. The design of the entire chip must of course take some account of the integrated hard core.[93] Even so, the hard core remains physically separate from the larger design. In addition, the core contributes functionality that, while essential to the device, is peripheral to the functionality of the proprietary design. Moreover, the hard core uses a standard interface to communicate with the remainder of the device.[94] In light of these facts, the proprietary device design should be considered a "separate and independent work"[95] under the terms of the GPL, and the device design (outside the hard core) should not be subject to the GPL copyleft obligations.

More complex situations, however, are likely to require a fact-based analysis of the degree of integration of the GPL-licensed hard core. For example, a hard core may not interface properly with the remainder of a device, and this may require the design of an entirely new interface.[96] The GPL may require that the source code of this

---

91. *See* CARBALLO, *supra* note 25, at 2; KEATING & BRICAUD, *supra* note 2, at 48 (discussing standard chip interfaces for connecting on-chip blocks, including I/O blocks); *id.* at 64 (discussing the need for standard interfaces for integrating hard cores).

92. In this example, as the core is licensed pursuant to the GPL, the fabless company would also receive the source HDL code of the core. Even so, integration of the core in hard rather than soft format may prove more efficient.

93. For example, the layout of the entire chip must leave physical space for the hard block, as well as account for its power, ground, and clock timing. *See* KEATING & BRICAUD, *supra* note 2, at 39 (stating that improperly placed hard cores "can cause blockage in the placement and routing of the entire chip"); *id.* at 220 (stating that the physical design of the chip must account for the "power and ground rings" and timing of the hard core). For a description of the insertion of clock and power networks into the device design, see CARBALLO, *supra* note 25, at 142.

94. The Free Software Foundation has stated that the use of communication channels typically used by distinct software programs shows that the two programs should be considered separate. *See GNU, FAQ*, *supra* note 76.

95. *GNU General Public License*, *supra* note 1, § 5.

96. *See* KEATING & BRICAUD, *supra* note 2, at 42.

interface be disclosed.[97] Integration of the hard core may require large-scale and significant changes to the power and timing of the device. Furthermore, certain issues may require changes to the architecture of the entire system.[98] Generally, to the extent that the GPL-licensed hard core provides peripheral functionality which uses standard interfaces and which does not demand internal changes to the remainder of the design of the device, the proprietary design will likely not be seen as a derivative work of the GPL core. On the other hand, to the extent the integration of the GPL core requires internal creative changes to the proprietary work, some part of the latter may be considered a derivative work and subject to the licensing conditions of the GPL.

Some open source semiconductor cores are licensed under the terms of the LGPL, most likely in an attempt to prevent copyleft licensing terms from attaching to the rest of the device design.[99] Based on the previous analysis, to the extent that a hard core remains physically separate and uses a specified interface, the GPL should in any event not require disclosure of works with which the hard core is integrated. Thus it is not clear whether the use of the LGPL for hard cores is necessary to limit application of the GPL copyleft provisions.

B. Soft Cores

The integration of a soft core into a device design presents a somewhat more complicated situation than the licensing of a hard core. A soft core simply presents the logical structure of a hardware circuit. This logical structure, however, can translate to any number of actual physical implementations. As explained in greater detail below, the exact expression of the soft core together with the remainder of the device design is in practice influenced by the other components of the device, as well as constraints imposed on the entire device design. In other words, the implementation of the design can both transform and be transformed by the soft core — creating a derivative work. Thus the use of GPL soft cores in device design presents a risk that at

---

97. The GPL definition of "Corresponding Source" expressly requires the disclosure of source code for the "interface definition files" associated with the work. *GNU General Public License*, *supra* note 1, § 1. *But see* Determann, *supra* note 86, at 1442 (stating that "interface modifications dictated by functional requirements do not support a classification as derivative works, as they may be relevant for the program's functionality, but do not significantly affect an author's reasonable interest in controlling adaptations of her creative works").

98. KEATING & BRICAUD, *supra* note 2, at 219 (describing interface problems that may require modification of the chip design, "perhaps even requiring changes to the architecture of one of the blocks or the entire system").

99. For example, OpenCores recommends that users employ the LGPL. *See What License Is Used for OpenCores?*, *supra* note 8.

least part of a proprietary design may be considered a derivative work of the soft core and therefore subject to the GPL obligations.

The technologically complex process of incorporating a soft core into a device requires the application of several stages of EDA software. The EDA software must determine the optimal physical placement of an extraordinarily large number of electronic components and ensure that the connecting wires do not interfere with each other.[100] Engineering teams place constraints on the size, power, and performance of the device as a whole, as well as imposing additional timing, power, and technical constraints with respect to each individual component.[101] In finding a solution to this three-dimensional jigsaw puzzle, EDA software typically manipulates eight or more layers of matematerial.[102] This complex process means that changes in the HDL source of a device or in the constraints imposed during the layout process can lead to very different physical configurations.[103] As a result, it is difficult to predict how the logical structure of a soft core would appear when integrated into the final three-dimensional design of a physical device, or how the integration of the soft core would itself influence the structure of the rest of the device. In other words, the device design incorporates the original soft core, but at the same time recasts that soft core into another form. In addition, the remainder of an HDL device design is itself transformed by the inclusion of a soft core in that design. Thus it is not difficult to see the device (or at least the part of the device affected by the integration of the soft core) as a "derivative work" of the core. This analysis means that the incorporation of a GPL-licensed soft core may subject at least part of the device design to the rights and obligations of the GPL.

Use of the LGPL may ameliorate some of these issues. As discussed above, and subject to certain conditions, the LGPL expressly allows for the combination of LGPL-licensed works with proprietary works, without requiring that the source code of the proprietary works be disclosed.[104] In order to qualify for this eased copyleft obligation, a proprietary work that makes use of "an interface provided by" the LGPL-licensed work must not be "based on" the LGPL-licensed

100. For a description of the intricacy of the layout process, see IBM PRIMER, *supra* note 17, at 19. In understanding the complexity of the process, it may be useful to compare device design to the compiling of the source code of a software program. Custom compilation of the entire Ubuntu Linux kernel, for example, which contains millions of lines of code, can take up to eight hours of compilation time. *See KernelCompile*, UBUNTU DOCUMENTATION (last updated Nov. 17, 2011, 7:51 AM), https://help.ubuntu.com/community/Kernel/Compile. In contrast, the physical design of a semiconductor device may demand tens of thousands of hours from a team of engineers. *See* BROWN & LINDEN, *supra* note 6, at 65 tbl. 3.1.

101. *See* CARBALLO, *supra* note 25, at 58–59, 89–90.

102. TURLEY, *supra* note 18, at 70.

103. For a discussion of the various factors and considerations, see *supra* text accompanying notes 24–25.

104. *See supra* text accompanying notes 73–74.

work.[105] Thus an independently developed soft core that is integrated into the device design through a standard interface should meet the criteria of the LGPL. However, a soft core that is specifically developed to add certain functionality to a particular device design may be considered "based on" that device design, and so would not meet the LGPL's requirements. In the same way, a soft core that does not make use of a specified interface but is otherwise integrated into the device design would not qualify. Thus an LGPL-licensed soft core may be integrated into a device design under certain circumstances without imposing copyleft obligations on the remainder of the design. Commercial entities taking this approach should take care to ensure that they comply with the other requirements of the LGPL.[106]

Another possibility for commercial entities interested in integrating open source soft cores would be to "harden" the GPL core separately from the remainder of the device. In other words, the layout of the soft core itself could be fixed in a GDSII file separately from the remainder of the device. This separate GDSII file could then be physically fixed into the entire design like a hard core. This use of "virtual" hard cores is sometimes used in the industry to increase design efficiency.[107] In this event, the problems with making use of the GPL soft cores would then reduce to the somewhat simpler issues discussed above with regard to hard cores.

C. Libraries

Another issue that arises in the semiconductor core context is the integration of the device design with proprietary "libraries." As noted above, commercially available EDA tools use libraries that describe the actual components that will be used in the actual fabrication process. Standard "cell" libraries, for example, provide designs for each logic gate to be used in a chip.[108] During the electronic design process, these libraries are used to explicate and give detail to an abstract logic design.[109] Both third-party providers and foundries offer libraries, the latter of which may do so as an incentive to attract custom-

105. *GNU Lesser General Public License*, *supra* note 1, § 0 (defining an "Application" as "any work that makes use of an interface provided by the Library, but which is not otherwise based on the Library"). Pursuant to the LGPL, a proprietary Application may be combined with an LGPL-licensed work to produce a "Combined Work," and the source code of the entire work may not need to be disclosed. *Id.*

106. Section 4(d)(0) of the LGPL, for example, would require providing the proprietary design under license terms "that permit[] the user to recombine or relink" the proprietary design with a modified version of the LGPL-licensed core. *Id.*

107. KEATING & BRICAUD, *supra* note 2, at 181.

108. *See* CARBALLO, *supra* note 25, at 59.

109. *See* TURLEY, *supra* note 18, at 54.

ers.[110] These libraries may be licensed under proprietary license terms that prohibit use of the libraries except in the manufacturing process of the providing foundry.[111]

The combination of an open source semiconductor core with a proprietary library will inevitably create a derivative work: the final product of the combination will be a GDSII file that incorporates both the design of the semiconductor core and the specifications of the library. In other words, the final device layout includes the core, but transforms and recasts that core into a different form by combining it together with a technology library. Thus under the GPL the final layout would be considered a derivative work of the core and would require that the entire layout (including the elements of the library inincorporated in the layout) be subject to its terms. This outcome may not be permitted under the terms of the proprietary library license.[112]

Use of the LGPL may address some of the issues raised by the integration of a technology library in a core. As discussed above, subject to certain conditions, the LGPL permits the combination or linking of proprietary programs together with LGPL-licensed works. Thus integrating a technology library together with a soft core may not trigger the copyleft obligations of the LGPL-licensed core. Again, commercial entities taking this approach should take care to ensure that they comply with the other requirements of the LGPL.[113]

It is possible that certain technology libraries would satisfy the "system library" exception of the GPL.[114] The source code of libraries

---

110. *See* HURTARTE, *supra* note 2, at 51; KUMAR, *supra* note 28, at 144; TURLEY, *supra* note 18, at 54.

111. *See, e.g.*, *Cadence and TSMC Library Distribution*, *supra* note 29.

112. Similarly, the integration of a proprietary core with an open-source library may create a derivative work of the open source library, which may not be permitted under the terms of the applicable licenses. The likelihood that a core will be seen as a derivative work of the library is increased to the extent that the design of the core is based on the functionality of the library. *See* CARBALLO, *supra* note 26, at 60–61 (providing examples of how core design may need to take account of the constraints of the library).

113. *See GNU Lesser General Public License*, *supra* note 1, § 4(d)(0). As noted above, the LGPL does not ease copyleft obligations to the extent that a work is "based on" the LGPL-licensed work. As such, use of the LGPL may not limit copyleft obligations to the extent that the design of the core is based on functionality provided by the library. *See supra* note 112.

114. Section 1 of the GPL provides that the source code of "System Libraries" is not included in the source code that must be disclosed. In the software context, this system library exception permits GPL-licensed programs to link with certain libraries that are not compatible with the GPL, such as standard libraries of common programming languages. *See* BRETT SMITH, FREE SOFTWARE FOUND., INC., A QUICK GUIDE TO GPLV3 (2007) http://www.gnu.org/licenses/quick-guide-gplv3.pdf; *see also* MEEKER, *supra* note 62, at 257. Section 1 of the GPL defines the term "System Libraries" as "anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form." *GNU General Public License*, *supra* note 1. A "Major Component" is generally a core component of the operating system or a compiler. The exact (and complex) definition of the term "Major Component" in § 1 of

satisfying that exception need not be disclosed in a distribution of the GPL-licensed work. The purpose of the system library exception is to exempt "core components of the operating system" as well as libraries that "users of the software can reasonably be expected to have" from the requirement to distribute source code.[115] The definition of a system library is relatively complex, and determining whether any specific technology library satisfies that definition is likely to be fact-intensive. For example, whether any specific library satisfies the definition depends on the details of how that library is distributed, since the GPL (somewhat opaquely) requires that a system library "be included in the normal form of packaging of a Major Component, but not actually form a part of that Major Component."[116] Furthermore, applying the system library exception to core libraries will also require some interpretation of terms originating in and more suitable to the software context. The GPL definition of "Major Component," for example, also includes a "compiler used to produce the work."[117] It is possible that EDA software that creates netlists and GDSII files from HDL code could be viewed as a "compiler."[118] If that were to be the case, technology libraries included with such EDA software could satisfy the system library exception.

VI. DISTRIBUTION OF PHYSICAL DEVICES

Are there any consequences to distributing an actual semiconductor device that incorporates a GPL-licensed core? Would the GPL, for example, require that the source code of such a device be disclosed? The Free Software Foundation has taken the position that the obligations imposed by the GPL will not reach the "physical hardware" created using GPL-licensed material.[119] In other words, the distribution of a physical semiconductor device — even if that chip were manufactured using GPL-licensed designs — would not require the concomitant distribution of the source code of that design.

This Article instead makes the counterintuitive argument that there is a risk that the requirements of the GPL *would* apply to the

---

the GPL is "a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it." *Id.*

115. *See* SMITH, *supra* note 114.

116. *GNU General Public License*, *supra* note 1, § 1. For an explanation of what constitutes a "Major Component," see *supra* note 114.

117. *GNU General Public License*, *supra* note 1, § 1.

118. *See supra* text accompanying note 30 for a comparison between software compilation and the process of creating a netlist.

119. *See, e.g.*, *GNU, FAQ*, *supra* note 76 ("Any material that can be copyrighted can be licensed under the GPL . . . . However, if someone used that information to create physical hardware, they would have no license obligations when distributing or selling that device: it falls outside the scope of copyright and thus the GPL itself.").

distribution of such a device. First, the Free Software Foundation's position is to some extent based on its understanding of the GPL as a license rather than a contract. Commentators, however, have disagreed with this position. Second, it is not clear that the Free Software Foundation's position is based on an accurate understanding of copyright law. Third, the Free Software Foundation's position is a consequence of the typical application of the GPL in the familiar context of software licensing — an area of law dominated by copyright law. This position, however, may not be the correct interpretation of the GPL in a context that also implicates additional intellectual property regimes expressly invoked by the GPL.

First, as noted above, the position of the Free Software Foundation seems to be based on its assumption that the GPL constitutes a license and not a contract.[120] According to this position, copyleft obligations are conditions that apply to the exercise of the GPL's license provisions — if the copyright license is not needed, then the conditions do not apply.[121] Thus since the manufacture of physical devices does not require the copyright license offered by the GPL, such activities are also not subject to the obligation to distribute source code. However, it is not clear whether the Free Software Foundation's position on this matter should be considered determinative. Commentators, for example, have disagreed with this position.[122] Other licensors using the GPL may take positions that differ from that of the Free Software Foundation and seek to enforce the terms of the GPL as a contract. Such licensors may assert that the GPL, as a contract, can impose contractual rights and obligations that would be beyond the reach of copyright law.

Commercial licensing arrangements typically include contractual provisions that reach beyond the rights granted by copyright law.

---

120. *See* FREE SOFTWARE FOUND., GPLV3 SECOND DISCUSSION DRAFT RATIONALE 22 n.77, *available at* http://www.gnu.org/licenses/quick-guide-gplv3.pdf (last visited Dec. 21, 2011) [hereinafter SECOND DRAFT RATIONALE] (stating that the GPL was "intentionally structured . . . as a unilateral grant of copyright permissions, the basic operation of which exists outside of any law of contract"); MEEKER, *supra* note 62, at 177. For a discussion of the differences between contracts and licenses in the context of the GPL, see ROSEN, *supra* note 81.

121. For example, the Free Software Foundation, referring to a previous draft of the GPL, has stated that the copyleft provisions of the GPL should extend only to "any modified version [of the licensed work] for which permission is necessary under copyright law." FSF, *Denationalization*, *supra* note 75. The Free Software Foundation's position that the distribution of hardware is not covered by copyright law and that, as such, the distribution of hardware should not trigger the copyleft obligations of the GPL is consistent with this approach. *See supra* note 119.

122. *See* MEEKER, *supra* note 62, at 225 (stating that commentators have expressed various reasons for believing that the GPL constitutes a contract, including the fact that some countries' laws do not permit a license that is not a contract, that the Uniform Commercial Code warranty disclaimers in § 11 apply only to contracts, and that "the circumstances surrounding acceptance of the terms of GPL may be sufficient to constitute acceptance of the document as a contract in most cases").

Commercial semiconductor core licenses, for example, often contain blanket restrictions on reverse engineering despite the fact that copyright law may permit reverse engineering under certain circumstances.[123] Furthermore, commercial licenses often contain contractual provisions such as warranty disclaimers, indemnification obligations, limitations of liability, and confidentiality restrictions.[124] All of these provisions fall outside the scope of copyright law, and their enforcement requires the application of contract law. Many of these contractual provisions could also govern the distribution of physical devices based on the licensed core. The GPL, if seen as a contract, could similarly be interpreted to impose contractual obligations that affect the rights and obligations of the parties with respect to the final manufactured product.

Second, it is not clear that the position of the Free Software Foundation reflects an accurate description of copyright law. An instructive parallel is the copyright granted to architectural plans. Several cases have held that the construction of a building may constitute publication (and thus infringement) of the architectural plans of that building.[125] A constructed building may also constitute a derivative work of such architectural plans.[126] In the same vein, the manufacture of a semiconductor device may constitute publication of the design on which such device was based, and the physical device may constitute a derivative work of that design. Thus the distribution of hardware based on a GPL-licensed design may be subject to the conditional license of the GPL.

Third, the GPL purports to grant rights under intellectual property regimes aside from copyright law — specifically, licenses under mask work and patent rights.[127] While copyright protection is limited to the specific expression of an idea, these other intellectual property regimes grant some protection to the idea itself. "Mask work" protection, for example, is a sui generis intellectual property right specifically created to protect the layout of semiconductor devices,

---

123. HURTARTE, *supra* note 2, at 93 (noting that core license agreements "may include a prohibition on reverse engineering, decompiling of computer code, [and] black box probing of the . . . product or disassembly").

124. *Id.* at 93, 96–99. Many other provisions that are typically included in semiconductor core license agreements have been singled out by the Free Software Foundation as clauses that are appropriate for contractual agreements but not for pure licenses. For example, earlier drafts of the GPL expressly listed common contract provisions that the Free Software Foundation considered inappropriate for a pure license such as the GPL. *See* SECOND DRAFT RATIONALE, *supra* note 120, at 20 n.70 (stating that provisions "typically found in license documents drafted from a contract-oriented perspective" are not compatible with the GPL).

125. *See generally* 1 NIMMER ON COPYRIGHT, *supra* note 88, § 4.12[C].

126. *Id.*

127. For a discussion of GPL's grant of an express mask work license, see *supra* text accompanying note 14. Section 11 of the GPL provides for express patent license provisions, which are discussed in more detail below.

and generally proscribes the unauthorized distribution of the physical devices based on a protected work. The United States Semiconductor Chip Protection Act of 1984 ("SCPA") prohibits the unauthorized manufacture, import, and distribution of semiconductor chips that embody the protected mask work.[128] In this respect, the SCPA follows the World Trade Organization Agreement on Trade-Related Aspects of Intellectual Property Rights ("TRIPS"), which provides the copyright holder with the exclusive right to incorporate a protected semiconductor layout in physical devices.[129]

The incorporation of an express mask work license in the GPL seems to extend the license (as well as the concomitant obligation to disclose source code) to the manufacture and distribution of physical devices based on the licensed work. Specifically, sections 5 and 6 of the GPL grant conditional licenses to convey "covered works" in any form, subject to the copyleft obligations imposed by the GPL.[130] "Covered works" are defined by the GPL to include "a work based on the Program"; this language is broad enough to include physical semiconductor devices manufactured from the specifications of the licensed core.[131] Thus the mask work rights expressly licensed pursuant

---

128. Semiconductor Chip Protection Act, 17 U.S.C. §§ 901–14 (2006). The SCPA provides that the owner of a mask work has the exclusive right to "reproduce the mask work by optical, electronic, or any other means." *Id.* § 905(1). The House Report on the Act clarifies that this prohibition includes the reproduction of the mask work in a semiconductor chip product. *See generally* Richard H. Stern, *Determining Liability for Infringement of Mask Work Rights Under the Semiconductor Chip Protection Act*, 70 MINN. L. REV. 271, 279 (1985). Similarly, § 905(2) of the SCPA grants the owner of the mask work the exclusive right to "import or distribute a semiconductor chip product in which the mask work is embodied." It is not clear whether the SCPA protects the HDL code or layouts used in the early stages of semiconductor device design. Several commentators have asserted that SCPA protection is limited only to misappropriation of the actual mask work used to create a semiconductor device. *See* Leon Radomsky, *Sixteen Years After the Passage of the U.S. Semiconductor Chip Protection Act: Is International Protection Working?*, 15 BERKELEY TECH. L.J. 1049, 1057–58 (2000). The laws of other jurisdictions as well as international treaties may provide protection to the earlier stages of semiconductor design. *See, e.g.*, *id.* at 1065–66 (summarizing the integrated circuit topography acts of several countries).

129. Article 36 of TRIPS extends protection to physical articles incorporating a protected design, providing that it shall be unlawful to import, sell, or distribute "an integrated circuit in which a protected layout design is incorporated." Agreement on Trade-Related Aspects of Intellectual Property Rights, art. 36, Apr. 15, 1994, 1869 U.N.T.S. 299, 33 I.L.M. 1197 (1994). Similarly, the applicable European Community directive provides that the exclusive rights granted to protected designs includes the "commercial exploitation or the importation for that purpose of a topography or of a semiconductor product manufactured by using the topography." Council Directive 87/54, art. 5(1)(b), 1987 O.J. (L 24) 36 (EEC).

130. *GNU General Public License*, *supra* note 1, §§ 5, 6.

131. *Id.* § 0; *see, e.g.*, *supra* text accompanying note 126 regarding whether a physical object can be considered a derivative work of a design. In a similar vein, even though the GPL does not expressly refer to a license to manufacture physical hardware based on a licensed design, such a license may be inferred. For example, cases in which architects have provided licenses to architectural plans have found that the contract also includes an implied license to build structures that embody those plans. *See* Nelson-Salabes, Inc. v. Morningside Dev., LLC, 284 F.3d 505, 516 (4th Cir. 2002) (surveying the existing case law, and stating that architects had generally been found to grant an implied license to construct buildings

to the GPL seem to allow for the manufacture of physical devices, but only subject to the condition that the licensee must also distribute the source code of that device. In the language of the Free Software Foundation, while it is possible that the distribution of physical devices "falls outside the scope of copyright," it does not fall outside the scope of the mask work rights also licensed under the GPL.[132] This interpretation of the GPL would require the distributor of physical devices incorporating GPL-licensed designs to distribute the source code of the licensed design incorporated in the device.

A similar argument can be made under the express patent provisions of the GPL. Section 11 of the GPL, for example, provides that each contributor to a licensed work generally grants users a non-exclusive patent license to "make, use, . . . modify and propagate" the distributed work.[133] This license is also subject to the conditions of the GPL — including the disclosure of source code together with any distribution — since section 8 of the GPL provides that violation of those conditions will mean the termination of this license.[134] As with the express mask work license provisions of the GPL, these provisions would seem to preclude limiting the obligations of the GPL to the activities permitted by the GPL copyright license, since the express patent license applies these same conditions to a larger universe of actions. Thus the express patent license provisions may extend the conditions of the GPL to distribution of physical devices based on the licensed work.

While the Free Software Foundation has opined that the distribution of hardware does not trigger the copyleft obligations of the GPL, this discussion has shown that neither the law nor the language of the GPL itself conclusively supports such a position. Furthermore, other licensors may take positions that differ from that of the Free Software Foundation and seek to enforce these differing interpretations of the

---

based on licensed designs "where the architects were hired for discrete tasks, with no indication of their further involvement in the project, and where they did not suggest . . . that use of the copyrighted material without their involvement was impermissible"). Similarly, the unrestricted distribution of a design may carry an implied patent license to manufacture devices based on the design. *See* Wang Labs., Inc. v. Mitsubishi Elecs. Am., Inc., 103 F.3d 1571, 1581–82 (Fed. Cir. 1997) (stating that Wang provided Mitsubishi with an implied patent license to manufacture semiconductor devices because it had also "provided designs, suggestions and samples" of such design without restriction).

132. *See supra* note 119.

133. *GNU General Public License*, *supra* note 1. The GPL contains other patent license provisions as well, and a similar argument can be made under those other provisions of the GPL. For example, the fifth paragraph of § 11 of the GPL may require the disclosure of source code if a covered work is conveyed while "knowingly relying on a patent license." *Id.*

134. *Id.* The first paragraph of § 8 of the GPL provides "You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify [the covered work] is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11)." *Id.*

GPL. In light of these conflicting interpretations, the obligations of a commercial licensee distributing hardware that includes a GPL-licensed design are not currently clear. Distributors of commercial devices that include GPL-licensed designs but do not also provide the source of those designs may risk being found in violation of the GPL.

VII. CONCLUSION

This Article has reviewed the application of the GPL to the licensing of semiconductor cores. A semiconductor core may resemble the standard software code typically licensed under the GPL. Even so, given the structure of the industry and the fact that cores are used to develop a range of expressions, the effect of the GPL's copyleft obligations may complicate standard commercial practice. Furthermore, the need to interpret provisions of the GPL that were originally intended for the standard software context makes the application of the GPL's provisions to semiconductor cores more difficult. Using cores licensed pursuant to the LGPL, as well as isolating incorporated core material using hard cores and standard interfaces, may mitigate or avoid these complications in certain situations.

A more practical long-term suggestion, however, may be to avoid licensing semiconductor cores under the GPL or other software licenses. The advantage of the GPL (as well as other copyleft licenses) is that it promises that improvements to open source material will be made available to the development community. Similar copyleft provisions, however, could also be incorporated in a license that takes account of the specific commercial needs of the fabless industry. Commentators have heavily criticized the proliferation of various open source licenses, noting that the wide variety of currently available licenses imposes burdens on licensors and users.[135] However, as the application of the GPL to semiconductor cores raises a number of complex issues, it may be appropriate to use a license tailored to the technological context of semiconductor cores. While the scope of copyleft obligations in any license typically raises complex issues of interpretation and analysis of technical detail, the application of a software license to a different technological context unnecessarily complicates these details.

A license customized for the commercial semiconductor core industry could directly address the issues raised in this Article.[136] Such a

135. *See, e.g.*, *Report of License Proliferation Committee and Draft FAQ*, OPEN SOURCE INITIATIVE, http://www.opensource.org/proliferation-report (last visited Dec. 21, 2011).

136. While "open hardware" licenses do exist, they do not take account of many of the complexities of the semiconductor device manufacturing process. For example, the TAPR Open Hardware License does not address the use of technology libraries, the incorporation of soft cores in a device design, or the use of independent contractors for parts of the design process. *See generally* Ackermann, *supra* note 15. In July 2011, the European Organization

license, for example, could contain provisions that expressly permit the combination of proprietary technology libraries with the licensed design. As described above, the use of proprietary technology libraries is a necessary part of the semiconductor device manufacturing process, especially as foundries may require the use of their own libraries in order to manufacture a device. Thus any license to be applied to semiconductor cores and used in a commercial context must contain provisions that permit the use and incorporation of technology libraries. In addition, a tailored license should clarify that under appropriate circumstances the use of subcontractors for the internal testing and development of the core would not trigger copyleft obligations. As noted above, the outsourcing of specific stages of semiconductor design is integral to the current structure and economics of the semiconductor industry, and any license that hopes to gain widespread acceptance in the industry should take account of these needs. A tailored license should also specify the circumstances under which the incorporation of a soft core into a larger device design would result in the application of copyleft obligations to the entire design.

The semiconductor industry has been moving further toward the use of independently developed cores to speed the creation of new devices and products. However, the need for robustly maintained and supported cores and the absence of clear rules and licenses appropriate for the industry's structure and practice have stymied the development of an open source ecosystem, which might otherwise have been a natural outgrowth of the use of independently developed cores. The development of a context-specific open source license may be the simplest way to clarify the applicable legal rules and encourage the commercial use of open source cores.

---

for Nuclear Research ("CERN") released version 1.1 of the CERN Open Hardware Licenses, but these issues are not addressed by that license either. *See CERN Open Hardware License*, OPEN HARDWARE REPOSITORY, http://www.ohwr.org/documents/88 (last visited Dec. 21, 2011).