Notre Dame Law Review

Volume 59 | Issue 2 Article 4

1-1-1984

Copyrightability of Object Code

James Canfield

NOTE

The Copyrightability of Object Code

> If the process were stopped at any one stage, would the [computer] program as it existed at that point be entitled to legal protection? This question, as the extensive literature proves, is a profound and troubling one, embracing not merely science and law, but philosophy as well.[1]

Recognizing the importance of the computer to American society, *Time* conferred its prestigious "Man of the Year" award on the computer in 1983, dubbing it "Machine of the Year."[2] The "Machine of the Year" is useless, however, without a computer program,[3] and meaningful legal protection for a computer program is not yet guaranteed.

Legal protection for computer programs includes three different types of protection: patent, trade secret, and copyright. Commentators have acclaimed each as the best method of legal protection.[4] This note examines copyright protection and focuses on object code[5] to determine if copyright offers *meaningful* legal protection for computer programs. As the quotation above suggests, an analysis of legal protection for computer programs involves three disciplines: science,[6] law, and philosophy.

---

1 Gemignani, *Legal Protection for Computer Software: The View from '79*, 7 RUTGERS J. COMP.,.TECH. & LAW 273 (1980).

2 Machine of the Year, TIME, Jan. 3, 1983, at 14.

3 *See* text accompanying notes 6-21 *infra*.

4 For an article in favor of patent protection, see Bender, *Computer Programs: Should They be Patentable?*, 68 COLUM. L. REV. 241, 248 (1968)("Although there are methods of protection other than patent laws, none of these methods meets as well as patent protection the needs of the computer industry."). For an article in favor of trade secret protection, see Nimtz, *Development of the Law of Computer Software Protection*, 61 J. PAT. OFF. SOC'Y 3, 25 (1979) ("[I]t is more likely that some form of trade secret protection will survive. This form of protection is currently the most widely used, the most reliable and . . . the residual form of protection when all other forms are either in doubt or fail."). For an article in favor of copyright protection, see Root, *Protecting Computer Software in the '80s: Practical Guidelines for Evolving Needs*, 8 RUTGERS COMP. & TECH. L.J. 205 (1981) ("[T]he combined evolution of the software market and copyright law has made copyright the preferred means of protection for software.").

5 *See* text accompanying notes 10-12 *infra*.

6 *See generally in order of relevance*, N. CHAPIN, COMPUTERS: A SYSTEMS APPROACH (1971); T.M. WALKER, INTRODUCTION TO COMPUTER SCIENCE: AN INTERDISCIPLINARY APPROACH (1972); J.A. O'BRIEN, COMPUTERS IN BUSINESS MANAGEMENT (3d ed. 1982); W. R. CORLISS, COMPUTERS (1973); J.A.N. LEE, THE ANATOMY OF A COMPILER (2d ed. 1974).

I. Scientific Background

A computer is a problem-solving machine that has three basic parts: (1) an input/output system, (2) a central processing unit (CPU), and (3) a main storage. As their names indicate, the input/output system inserts and transfers information in to and out of a computer, the central processing unit controls the translation and execution of instructions, and the main storage stores information for retrieval later. These three parts are referred to as "hardware." Computer hardware by itself, however, cannot solve any problems. A computer requires a program to solve problems; these programs are referred to as "software."

A computer program is "a set of statements or instructions to be used directly or indirectly in a computer in order to bring about a certain result."[7] Computer programs fall into two general categories: programs that supervise the CPU (called systems software) and programs that supervise solving particular problems (called application software). Regardless of which category the program falls into, the program is the end result of a two-part process involving: (A) developing an algorithm; and (B) translating the algorithm into either a computer language or machine language program.

A. *Algorithm Development*

Before a computer can be used to solve a particular problem, a programmer must determine the *method* for solving the problem. This method for solving the problem is known as an "algorithm." For example, if the problem is: "How do I get a date for Saturday night?," then the algorithm for one possible solution would be:

1. List all the available members of the opposite sex I know.
2. Order them according to my desire to date them.
3. Contact the highest ranked.
4. Ask for a date on Saturday night.
5. If the first declines, then ask the second.
6. Repeat until someone accepts or the list is finished.
7. If someone accepts, then I have my date.

So, an algorithm is simply a universal, detailed sequence for solving a particular problem. Another example of an algorithm, using a problem more appropriate for a computer, i.e., "what is the value for $x^4 + 2$," is:

7 17 U.S.C. § 101 (1976), *amended by* 17 U.S.C. § 101 (1980). The 1980 Amendment added this computer program definition.

1. get x
2. multiply x by itself
3. multiply the value in (2) by x
4. multiply the value in (3) by x
5. add 1 to the value in (4)
6. add 1 to the value in (5).

Algorithms of the type presented above (English-language representation) are only the first step in developing a method for solving a problem. For more complex algorithms, the English language becomes an unsatisfactory means of representing the method for solving the problem. For example, the repetitious process called for in steps (5) and (6) of the first algorithm and steps (2) through (6) of the second algorithm can be better represented graphically.

Therefore, the second step in developing a method for solving a problem is to replace the English-language representation of the algorithm with a "flow-chart" representation of the algorithm. The flow chart representing the "Saturday date" algorithm is given in Figure 1, p. 415. The flow chart for the "$x^4 + 2$" algorithm is given in Figure 2, p. 416.

After programmers have determined a method of solution for a particular problem, they can then use a computer to solve the problem, provided they can make the computer execute the steps in the flow chart. To do this the flow chart, or the algorithm, must be translated into a program the machine can execute.

B. *Translation*

Before discussing how a computer executes the steps in a flow chart, it is necessary to understand how the machine stores data. A computer can only represent data by either the presence or absence of a "signal" at a certain point within the machine.[8] Because the computer can only represent information through the presence or absence of a signal, it operates on a *binary* system. The physical representation of the presence or absence of a signal is translated into a two digit number system, where 0 and 1 represent the presence or absence of a signal. By grouping the 0's and 1's together into a number sequence (and necessarily, therefore, grouping the signals together within the machine), a code is created which can represent all Arabic numerals and all letters of the English language. The code represents both data and instructions for the machine. This code, the

8 The signal can take the form of open or closed switches in the machine or the presence or the absence of an electrical charge on a magnetic disk.

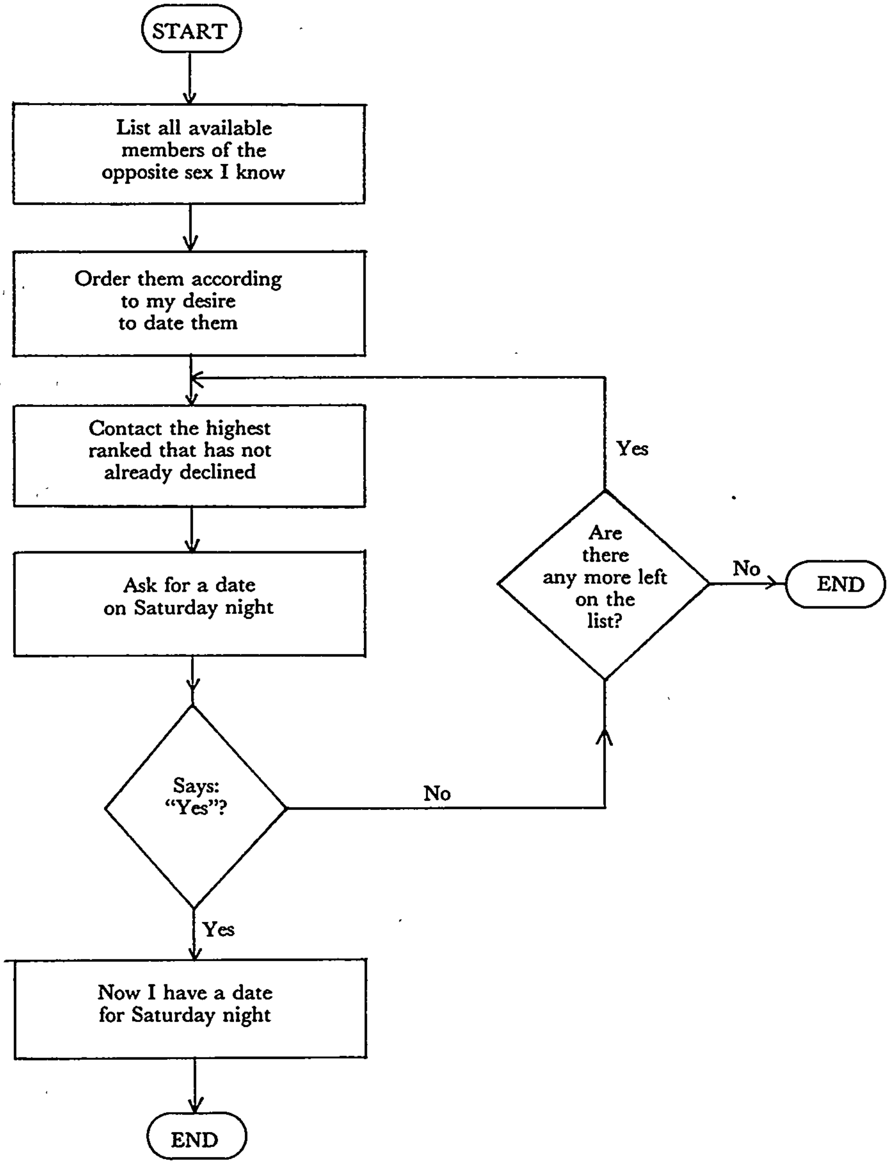

FIGURE 1. Flow chart representation for the "Saturday date" algorithm.

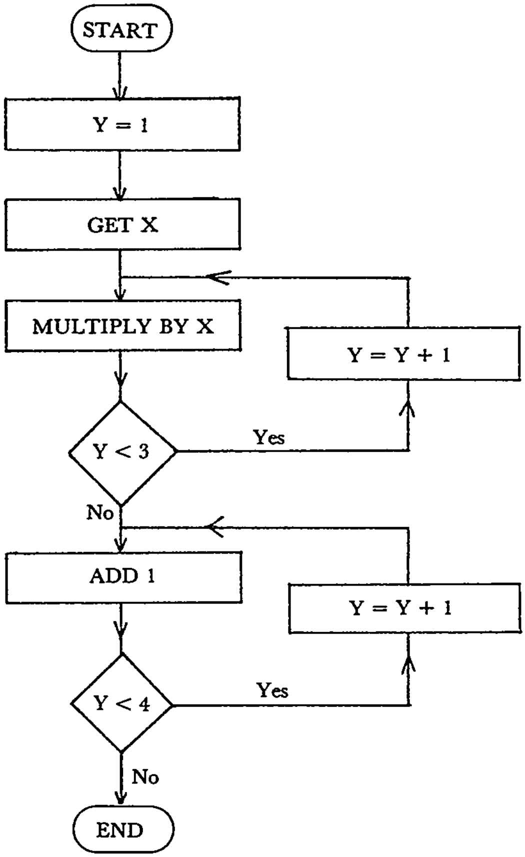

FIGURE 2. Flow chart representation for the "$x^4 + 2$" algorithm.

object code, also defines the exact location within the computer's circuitry of the signals which the code represents. The object code, consisting of nothing but 0's and 1's, is also referred to as machine language. Each computer has a unique machine language because each has a unique set of locations for the signals which represent each data item and each instruction.[9]

Returning to the problem of getting the computer to execute the steps in our flow chart, we can now discuss computer programming. Programming is the process of converting a flow chart representation of an algorithm into steps in a computer programming language or a machine language.

Computer programming languages such as BASIC, FORTRAN, and COBOL, are referred to as "high-level languages" because they more closely resemble English. In fact, BASIC has been called an English dialect.[10] Machine languages are referred to as "low-level languages." An example of FORTRAN is "X**4 + 2," which is the computer programming language expression for the flow chart in Figure 2. Note how simple this representation is, while the flow chart is complex. These high-level languages, referred to as source codes, are just shorthand codes for more complex steps. Primarily because programming in machine language is "slow, difficult, and error-prone,"[11] programming is done in computer programming languages.

Ultimately, for the computer to execute the FORTRAN program, it must translate the high-level computer programming language (source code) into low-level machine language (object code). A compiler program translates source code into object code. The compiler program is stored in the computer's main storage and controlled by the CPU.

In summary, the entire translation from source code to execution takes place as follows:

1. Source code is inputed into the computer.

9 Each *non-similarly designed* computer has a unique set of locations. All Apple II's, for example, have the same machine language since they are of similar design.

10 Stern, *Another Look at Copyright Protection of Software: Did the 1980 Act Do Anything for Object Code?*, 3 COMPUTER L.J. 1, 2 (1981).

11 There are six reasons for a programmer to choose source code over object code when writing a program: (1) speed; (2) ease of debugging; (3) inefficiency of the object language, i.e., inexperienced programmers will be less efficient than a compiler program (that program which translates source code into object code) if they write directly in object code; (4) efficient use of storage, i.e., again inexperienced programmers will be less efficient than a compiler program if they write directly in object code; (5) easier documentation; and (6) ease of learning. *See* CHAPIN, *supra* note 6, at 348-350.

2. The CPU copies the source code and places it in the computer's memory.
3. The CPU's control unit removes the compiler program from memory and places it in the working unit.
4. The control unit takes the source code from memory and gives it to the compiler program as data.
5. The compiler translates the source code into object code.
6. The object code then controls the execution of the problem.

In a variation on this *one time* translation process, object code can be etched on chips called "Read Only Memory" (ROM)[12] and *permanently* stored within the computer as a physical component.

In summary, computer programs direct a computer how to solve a problem. A programmer usually writes the program in source code, which is then translated into object code. The object code then directs the computer's execution of the steps in the program until a problem's solution is reached.

II. State of the Law

A. *Source Code versus Object Code*

As the technical information above explains, a computer program has two phases: source code and object code. Courts generally consider source code to be copyrightable,[13] which is not surprising. Source code is clearly a writing "written by a human author exercising the usual skills of human authorship, i.e., the selection of a particular mode of expression in a generally accepted language and using rules of syntax and grammar which insure intelligibility."[14]

To establish legal protection for the source code through copyright one must:[15]

1. Place a copyright notice at many places within the program and on the disk that contains the program.[16]
2. Fill out Copyright Office Form TX.

12 ROM's are semiconducting silicon chips that by their very construction have object code encoded into them. They are a permanent piece placed within the computer and have a fixed memory that is not changeable.

13 The state of the law is fairly clear regarding the copyrightability of source code. As the United States District Court for the Eastern District of Pennsylvania said in Apple Computer, Inc. v. Franklin Computer Corp., 545 F. Supp. 812, 822 n.15 (E.D. Penn. 1982), *rev'd on other grounds*, 714 F.2d 1240 (3d Cir. 1983): "Programs written in source codes are generally conceded to be copyrightable."

14 Nimtz, *Computer Programs*, in CURRENT DEVELOPMENTS IN PATENT LAW 1982, 143 PATENTS, COPYRIGHTS, TRADEMARKS AND LITERARY PROPERTY COURSE HANDBOOK SERIES, 141, 148 (PLI) (1982).

15 D. REMER, LEGAL CARE FOR YOUR SOFTWARE 26-34 (1982).

16 An example of a sufficient notice is: (c) 1984 James Canfield.

3. File Form TX with the Copyright Office along with the filing fee and a copy of the source code.[17]

After fulfilling these three steps, the source code has full copyright protection, and the copyright holder can sue another who uses the program without permission.[18] If the copyright holder proves infringement, then a court can award damages[19] or issue an injunction to stop the unauthorized use.[20] However, full copyright protection for source code does *not* fully protect the computer program from use by another.

To fully protect a computer program from use by another, *both* source and object code must be protected. Copyrighting the source code alone is insufficient because a dedicated programming expert can deduce the underlying object code from a copy of the source code. If the *object* code is not also protected, then the expert can use the deduced object code to run a computer. Similarly, if the object code in the form of an ROM is not protected, then, since an expert can examine the physical construction of the ROM with a microscope and "read" the object code etched therein, the computer program can be copied.

Consequently, one company can invest millions of dollars developing and translating algorithms into marketable, copyrightable source code,[21] yet another company can duplicate the first company's program by deducing, at a cost in only the *thousands* of dollars, the object code from either the copyrighted source code or the ROM. Therefore, while the source code is copyrightable, unless the object code is also protected, copyright protection for computer programs is practically useless.

B. *Uncertainty over Object Code*

A court first considered whether object code is copyrightable in *Data Cash Systems, Inc. v. JS&A Group, Inc.*[22] In *Data Cash* the plaintiff

17 A copy of the source code is the first and last twenty-five pages of the program. If one files object code, the Copyright Office will accept the registration under the "rule of doubt," meaning that the Copyright Office will accept it, but they cannot determine if it is really copyrightable. *See* REMER, *supra* note 15, at 34.

18 17 U.S.C. § 501 (1976).

19 The copyright holder can recover the amount lost due to the infringement or the profits made by the infringers. The court can award statutory damages, from $100-$50,000, even if one cannot prove that the infringers made a profit. 17 U.S.C. § 504 (1976). One can also recover court costs and, at the court's discretion, attorney's fees. 17 U.S.C. § 505 (1976).

20 17 U.S.C. § 502 (1976).

21 *See Computer Rights Stir Legal Snarl*, N.Y. Times, July 5, 1983, at 1, col. 6.

22 480 F. Supp. 1063 (N.D. Ill. 1979), *aff'd on other grounds*, 628 F.2d 1038 (7th Cir. 1980).

copyrighted a computer program, in source code, for use in a handheld computerized chess game. The manufacturer encrypted the program, in object code, into an ROM and then placed it in the game. The defendants produced a handheld chess game of their own containing an ROM they constructed after "reading" the plaintiff's ROM.

Both parties assumed the plaintiff's ROM was a "copy" of the plaintiff's copyrighted computer program for the game. However, the district court granted the defendants summary judgment, saying: "While the ROM is the mechanical embodiment of the source program, it is not a 'copy' of it."[23]

The court found that an ROM was not a "copy" of the source code under both the common law and the 1909 Copyright Act. To support their common law view, the court analogized to cases holding that a building is not a copy of the architectural plans upon which the building is based. Since only similar technical writings can be "copies" of architectural plans, the court said that at common law a "copy" of a computer program can only be another computer program in its *flow chart* or *source phase* since these are "comparable technical writings."[24]

To support their view that the ROM is not a "copy" of the source code under the 1909 Copyright Act, the court cited *White-Smith Music Publishing Co. v. Apollo Co.*[25] In *White-Smith*, the Supreme Court held that a piano roll was not a "copy" of the musical composition recorded thereon since "[i]n no sense can musical sounds which reach us through the sense of hearing be said to be copies as that term is generally understood."[26] Analogizing to *White-Smith*, the *Data Cash* court found that since one cannot see and read an ROM with the naked eye, it is not a copy under the 1909 Act.[27] The Court of Appeals for the Seventh Circuit affirmed *Data Cash*'s result[28] but did not consider the district court's position that an ROM is not a copy of the source code since neither side had contested the issue on appeal.[29]

In *Tandy Corp. v. Personal Micro Computers, Inc.*,[30] the court took a

23 480 F. Supp. at 1068.

24 *Id.*

25 209 U.S. 1 (1908).

26 *Id.* at 17.

27 480 F. Supp. at 1069.

28 628 F.2d 1038 (7th Cir. 1980).

29 *Id.* at 1041.

30 524 F. Supp. 171 (N.D. Cal. 1981).

position on the issue opposite to the *Data Cash* position. The plaintiff in *Tandy Corp.* had a home computer with an ROM that controlled the computer's translation of input data from high-level language to low-level machine language. The plaintiff claimed that the defendant had copied plaintiff's translation program. The defendant argued that the plaintiff's ROM chips were not copies of the original computer program and, therefore, defendant's ROM chips, which *were* copies of plaintiff's ROM chips, did not infringe the copyright covering the original program.

The court found for the plaintiff, holding that a computer program fixed in the form of an ROM is a "copy" of the original copyrighted program and therefore protected by the copyright laws. The court cited 17 U.S.C. § 102(a) which states that works can be fixed in "any tangible medium of expression, now known or later developed, from which they can be perceived, reproduced, or otherwise communicated, either directly or with the aid of a machine or device." The court said that imprinting a computer program into a silicon chip easily fell within this definition.[31]

In *GCA Corp. v. Chance*[32] the court followed *Tandy Corp.* and rejected *Data Cash*. In *GCA Corp.*, the plaintiff copyrighted its source code, but not its object code, for certain computer programs. The defendant contended that copyrighting only the source code does not protect the object code. The court said that because the object code encrypts the copyrighted source code, the two should be treated as one. Therefore, according to the court, the copyright of the source code protects both.[33]

In several cases dealing with video games the courts also followed the *Tandy Corp.* decision.[34] For example, in *Williams Electronics,*

31 *Id.* at 173.

32 217 U.S.P.Q. (BNA) 718 (N.D. Cal. 1982).

33 *Id.* at 720.

34 In Midway Mfg. Co. v. Artic International, Inc. 547 F. Supp. 999, 1008 (N.D. Ill. 1982), the district court said: "[T]he copyrighted material is recorded in the ROMs and may be reproduced with the aid of the microprocessor. While the technology of the reproduction is different than that encountered with videotape, Congress has allowed for that eventuality by allowing for fixation in mediums of expression either 'now known or later developed'." The Court of Appeals for the Seventh Circuit affirmed, 704 F.2d 1009, 1012 (1983), concluding that video games are copyrightable as audiovisual works. *See also* Midway Mfg. Co. v. Strohon, 564 F. Supp. 741, 746 (N.D. Ill. 1983) (The defendant does not dispute the "now well established" proposition that a video game's images and sounds qualify for copyright protection as an audiovisual work.); Midway Mfg. Co. v. Dirkschneider, 543 F. Supp. 466, 480 (D. Neb. 1981) ("[I]t is clear that the plaintiff's audiovisual works are fixed in the printed circuit boards.").

Inc. v. Artic International, Inc.,[35] the defendant copied, and sold in its own kits, the computer program for the video game DEFENDER. When sued for infringement, the defendant relied upon the *Data Cash* district court opinion. The court found for the plaintiff, saying that to allow duplication of a copyrighted computer program etched on a silicon chip would create an "unlimited loophole" for copyright infringement.[36]

Therefore, despite an initial finding in *Data Cash* that the object code is not a "copy" of the source code and not protected under the copyright laws, subsequent decisions rejected *Data Cash* and asserted that the object code is indeed a "copy" of the source code and is protected by the copyright laws. These decisions asserting that the object code is a "copy" of the source code are the better reasoned. Object code, when not initially written as object code, is simply the machine-translated version of source code. Admittedly, most people would find the translation unintelligible. But, a book translated into braille is no less a "copy" because it is translated into a form that is unintelligible to most people.

A more disturbing problem arises, however, in calling object code a *copyrightable* "copy" of the source code, because the Copyright Act requires "communication" and "expression" and object code is not always directed at a human audience. A court first confronted this problem in *Apple Computer, Inc. v. Franklin Computer Corp.* (*Franklin I*).[37]

In *Franklin I*, the plaintiff sued for a preliminary injunction restraining the defendant from infringing on the plaintiff's copyrights on fourteen computer programs written in object code. The copyright holder sued after the defendant built a personal computer that could run all of the plaintiff's programs. The court concluded "that there is some doubt as to the copyrightability of the programs."[38] Accordingly, the court denied the motion for a preliminary injunction since the plaintiff was unable to show a reasonable probability of success on the merits.

By focusing on the "audience" for a computer program, the *Franklin I* court confronted a problem that earlier cases ignored, i.e., whether the program in its object code phase always comes within the *scope* of the Copyright Act. The Copyright Act protects only

35 685 F.2d 870 (3d Cir. 1982).

36 *Id.* at 877.

37 545 F. Supp. 812 (E.D. Penn. 1982), *rev'd*, 714 F.2d 1240 (3d Cir. 1983) (*Franklin II*).

38 545 F. Supp. at 812.

"original *works of authorship* fixed in any tangible *medium of expression*, now known or later developed, from which they can be perceived, reproduced, *or otherwise communicated*, either directly or with the aid of a machine or device"[39] (emphasis supplied). The Copyright Act, therefore, essentially sets two requirements for a work to receive copyright protection: authorship and expression. One must remember, however, that Congress exercised a constitutional grant of power in the Copyright Act. The Constitution gives Congress the power "[t]o promote the Progress of Science and useful Arts, by securing for limited Times to Authors . . . the exclusive Right to their respective Writings."[40] Currently, Congress has not fully exercised this constitutional grant of power.[41] Any exercise of this power, however, must come within the scope of the general grant of power. Therefore, the philosophical questions "Who is an 'author'?" and "What is a 'writing'?" underlie any discussion of copyright protection.

III. Philosophy

A. *Who Is An Author?*

The United States Supreme Court has defined an author as "he to whom anything owes its origin; originator; maker."[42] A programmer writing a program in source code easily fits this definition, but object code, when translated from source code by a compiler pro-

---

39 17 U.S.C. § 102(a) (1976).

40 U.S. CONST. art. I, § 8.

41 *See* H.R. REP. NO. 94-1476, 94th Cong., 2d Sess. 1 (1976) *quoted in* Kernochan, *Subject Matter of Copyright; Rights in General; Rights in Relation to Computers*, in CURRENT DEVELOPMENTS IN COPYRIGHT LAW 1977, 81 PATENTS, COPYRIGHTS, TRADEMARKS AND LITERARY PROPERTY COURSE HANDBOOK SERIES 49 (PLI) (1977). The report states:

> In using the phrase 'original works of authorship,' rather than 'all the writings of an author' now in section 4 of the statute, the committee's purpose is to avoid exhausting the constitutional power of Congress to legislate in this field, and to eliminate the uncertainties arising from the latter phrase. Since the present statutory language is substantially the same as the empowering language of the Constitution, a recurring question has been whether the statutory and the constitutional provisions are co-extensive. If so, the courts would be faced with the alternative of holding copyrightable something that Congress clearly did not intend to protect, or of holding constitutionally incapable of copyright something that Congress might one day want to protect. To avoid these equally undesirable results, the courts have indicated that 'all the writings of an author' under the present statute is narrower in scope than the 'writing' of 'authors' referred to in the Constitution. The bill avoids this dilemma by using a different phrase—'original works of authorship'—in characterizing the general subject matter of statutory copyright protection.

H.R. REP. NO. 94-1476, at 51.

42 Burrow-Giles Lithographic Co. v. Sarony, 111 U.S. 53, 58 (1884).

gram,[43] appears authorless. A plaintiff can convincingly argue, however, that the compiler's translation of source into object code occurs on a predictable one-to-one basis and therefore preserves the programmer's authorship.[44]

Object code encrypted into an ROM presents a greater problem since it appears to have been "built" rather than "authored." In *Franklin I*, the court correctly suggested that an encoded ROM might be more aptly described as a "pictorial three-dimensional object" rather than as a "literary work," thereby discounting the authorship concept normally associated with literary works.[45] This, however, would not mean that the ROM is not copyrightable because three dimensional works of art *are* entitled to copyright protection.[46]

B. *What Is A Writing?*

In *Burrow-Giles Lithographic Co. v. Sarony*,[47] the Supreme Court defined writing as "the literary productions of . . . authors . . . includ[ing] all forms of writing, printing, engraving, etching, &c., by which the ideas in the mind of the author are given visible expression."[48] In a later case the Supreme Court, in dicta, extended this definition to include "any physical rendering of the fruits of creative intellectual or aesthetic labor."[49] Certainly, both source code and object code fit this later definition of "writing." However, not everyone accepts this broad definition of "writing." Two more restrictive tests for what constitutes a writing are the court's in *Franklin I* and Professor Nimmer's. Of the two, the *Franklin I* test is stricter.

In his treatise on copyright, Professor Nimmer states: "If a work contains a modicum of intellectual labor, and is embodied in tangible form, it would seem that it may constitute a writing if it is perceptive to any of the five senses."[50] Since source code can easily be read, it easily passes Nimmer's test. Arguably, object code also passes Nimmer's test since it *can* be read, although only by experts familiar with the binary code. The fact that meaningless code words have

---

43 Of course, if the program is written directly in object code by the programmer, then object code would have been authored.

44 545 F. Supp. at 822.

45 *Id.* In *Franklin II*, 714 F.2d at 1249, the Court of Appeals for the Third Circuit asserted that a computer program in object code *is* a "literary work." The *Franklin II* court, however, erred in not considering whether a "literary work" requires a human audience.

46 1 M. NIMMER, NIMMER ON COPYRIGHT § 2.08[B] (1982).

47 111 U.S. 53 (1884).

48 *Id.* at 58.

49 Goldstein v. California, 412 U.S. 546, 561 (1973).

50 1 NIMMER, *supra* note 46, at § 1.08[B] (footnotes omitted).

been afforded copyright protection supports Nimmer's position.[51]

The court in *Franklin I*, however, proposed a stricter test, requiring that whatever the form of expression or the medium used, the question must be whether the expression is directed to a human audience.[52] While source code and meaningless code words might fit this test, object code may not. Although a trained programmer may be able to read object code, it is usually directed solely to a machine.

Object code and computer hardware interaction (executed object code), however, can result in a perceivable *output* directed at a human audience (e.g., video games and print-outs). One commentator has argued, though, that since the programmer normally wants to protect the unexecuted object code, this code, not its output, must qualify as a writing.[53] This argument, however, simplifies the problem too much. If the program's output is irrelevant, then object code would certainly be copyrightable. Essentially, unexecuted object code is no different from "unexecuted" magnetic tape in a cassette. Yet the tape is copyrightable because it can be perceived, albeit with the aid of a machine. Object code creates doubt as to its copyrightability precisely because even when executed it may not be perceivable by a human audience.

The *Franklin I* court drew this distinction between object code that creates a perceivable output and object code that creates an unperceivable output. The court stated:

> If the concept of "language" means anything, it means an ability to create human interaction. It is the fixed expression of this that the copyright law protects, and only this. To go beyond the bounds of this protection would be ultimately to provide copyright protection to the programs created by a computer to run other computers. With that, we step into the world of Gulliver where horses are "human" because they speak a language that sounds remarkably like the one humans use.[54]

The *Franklin I* court, therefore, supported copyright protection for video game programs in the object code phase. The court stated that a video game program's purpose was to generate a perceivable image and to thereby attract a human audience and that such a purpose and goal satisfied "conventional expectations of expression."[55]

A Congressional commission confronted the issue of whether ob-

51 Reiss v. National Quotation Bureau, Inc., 276 F. 717 (S.D.N.Y. 1921).
52 545 F. Supp. at 824.
53 D. BROOKS, COMPUTER PROGRAMS & DATA BASES, 134 n.6 (1981).
54 545 F. Supp. at 825.
55 *Id.*

ject code is a "writing" and, therefore, acceptable material for copyright protection long before the issue was ever litigated. In 1974, Congress created the Commission on New Technological Uses of Copyrighted Works (CONTU) to study the use of copyrighted works in conjunction with computers. The commission members split eleven to three in favor of object code being considered a "writing." The commission, however, did not consider the problem of object code encrypted into an ROM.[56] The majority said:

> Because [programs] are used in conjunction with machines there has not been universal agreement about the propriety of copyright protection. Programs should no more be considered parts of machines than videotapes should be considered parts of projectors or phonorecords parts of sound reproduction equipment. All three types of works are *capable* of communicating with humans. . . ."[57]

In a concurrence, Commissioner Nimmer recommended limiting copyright protection for object code to only those programs whose *output* would qualify for copyright protection. Nimmer said:

> A program designed for a computer game would be copyrightable because the output would itself constitute an audiovisual work. On the other hand, programs which control the heating and air-conditioning in a building, or which determine the flow of fuel in an engine, or which control traffic signals would not be eligible for copyright because their operations do not result in copyrightable works. . . .[58]

Commissioner Hersey dissented, arguing that object code should not be considered a "writing" since it only communicates with the machine.[59]

The CONTU majority, therefore, reached a different conclusion from the *Franklin I* court concerning the copyrightability of object code. CONTU asserted that object code is a "writing" and within

---

56 NATIONAL COMMISSION ON NEW TECHNOLOGICAL USES OF COPYRIGHTED WORKS, FINAL REPORT 79 (1978). Because CONTU did not consider ROM's, one might argue that the CONTU report is irrelevant in the context of litigation involving ROM's. CONTU is relevant, however, because CONTU confronted the underlying issues, as is apparent in the opinions of the majority, Commissioner Nimmer, and Commissioner Hersey. Whether CONTU is relevant as legislative history is a separate question discussed in the text accompanying notes 72-74 *infra*.

57 *Id*. at 52.

58 *Id*. at 27.

59 Commissioner Hersey said: "[T]he direct product of a computer program is a series of electronic impulses which operate a computer; the 'writing' of the author is spent in the labor of the machine. . . . The computer program communicates, if at all, only with a machine." *Id.* at 73.

the scope of copyright protection, while the *Franklin I* court asserted that object code is within the scope of copyright protection *only* if the expression is directed at a human audience.

Since *Franklin I*, three courts have confronted the copyrightability of object code outside the video game context. In *Hubco Data Products Corp. v. Management Assistance Inc.*,[60] Management Assistance Inc. (MAI) alleged that Hubco modified computer operating systems and sold software that infringed MAI's operating system's object code. Hubco contended that object code is not eligible for copyright protection because it is a machine process that is not communicated to others.[61] The court conceded that *Franklin I* provided authority for Hubco's position but noted that the CONTU report and *Williams*[62] provided authority against Hubco. The court termed *Williams* and the CONTU report "well reasoned"[63] and, following their lead, found that Hubco had infringed on MAI's programs.

In *Apple Computer, Inc. v. Formula International, Inc.*,[64] Apple sued Formula for copyright infringement. Formula sold a personal computer, Pineapple, that Apple alleged contained ROM's which were virtually identical copies of Apple's ROM's. Formula had conceded that some computer programs are validly copyrightable.[65] Therefore, the primary issue was whether *all* computer programs including ROM's are copyrightable. Basing its decision upon the CONTU report and public policy,[66] the *Formula* court rejected Formula's dis-

60 1983 COPYRIGHT LAW REPORTER (CCH) ¶ 25,529.

61 As stated in Hubco's memorandum, quoted by the court:

> These operating system programs are never intended to be "perceived" by the recipient. In their function, they are transparent; they are not visible. In operation, they become an integral part of the machine. They control procedures and process. They establish the limits of the system, the method of operation. But, in and of themselves, these operating system programs are invisible. The user never preceives (sic), or views these codes; the user is aware only of their result in the functioning of his machine.

Id. at 18,105.

62 *See* text accompanying notes 35-36 *supra*.

63 1983 COPYRIGHT LAW REPORTER (CCH) at 18,105.

64 562 F. Supp. 775 (C.D. Cal. 1983).

65 Formula conceded:

> Some computer programs are validly copyrightable, i.e., those which contain expression. (Those which are not copyrightable, say Defendants, are those integral to the operation of the machine which do not directly produce visual communication with the user of the machine. All five items in issue are of the latter type.)(parenthetical comment by the court).

Id. at 779.

66 During its discussion of public policy, the *Formula* court said that it is up to Congress to determine if copyright, or a new form of protection, is best for the new technology. *Id.* at 783. An example of Congress at work in this area is the Semiconductor Chip Protection Act of

tinction between computer programs that directly produce visual communication with the user and those that do not.

The *Formula* court also distinguished and questioned *Franklin I*. While *Franklin I* and *Formula* may be distinguishable, the court erred in questioning *Franklin*'s validity simply because of *Williams*. As stated earlier, *Williams* involves a video game ROM. *Williams*, and all other video game cases, should be considered separately because the ROM's involved produce a visual output directed at a human audience. To cite as authority the holding from a video game case in a case involving a ROM with no such visual output inherently begs the question because these cases involve different programs. Even if the court does not decide the issue on the basis of "audiovisual work,"[67] a video game ROM does produce a perceivable output and, therefore, does not present the difficult question of the copyrightability of ROM's that produce unperceivable output.

In *Franklin II*,[68] the Court of Appeals for the Third Circuit reversed *Franklin I*, finding (1) no distinction between source code and object code and (2) that computer programs embedded on an ROM satisfy the Copyright Act's fixation requirements. The *Franklin II* court cites *Williams* as authority for both holdings. While it was correct for the court to cite *Williams* and other video game cases as authority for the proposition that ROM's satisfy the fixation requirements (as three of *Franklin II's* four cases cited on this point are), it was a mistake for the court, just as it was a mistake for the *Formula* court, to cite *Williams* as authority for finding no distinction between source code and object code. Worse, the court of appeals in

---

1983 which has been proposed in both houses. Regarding S. 1201, the Senate version of the bill, see 26 PAT. TRADEMARK & COPYRIGHT J. (BNA) 33, 46, (1983) and 27 PAT. TRADEMARK & COPYRIGHT J. (BNA) 72, 97 (1983) (bill was marked-up and referred out of committee); and regarding H.R. 1028, the House version of the bill, see 25 PAT. TRADEMARK & COPYRIGHT J. (BNA) 333, 341 (1983). The Act is designed to stop "chip piracy" by extending, for ten years, copyright protection to the imprinted design patterns on the chips.

Certainly, semiconductor chips affect our lives. As Senator Mathias stated:

> The chip is in the home, making dinner in the microwave oven, setting the thermostat and tuning the radio; it is in the supermarket, adding up our purchases; it is in the car, controlling fuel consumption; it is in the hospital, helping doctors diagnose disease; it is in the schools, instructing our children; and it is in the office, doing the typing, the recordkeeping, and almost everything else.

26 PAT. TRADEMARK & COPYRIGHT J. (BNA) 46 (1983). Yet, there is not universal agreement that copyright protection is appropriate for chips. As the counsel to the United States Copyright Office indicates, the relationship between the copyright in the chips and the copyright in the works of authorship embodied in the chips is unclear. 26 PAT. TRADEMARK & COPYRIGHT J. (BNA) 96 (1983).

67 *See Strohon*, 564 F. Supp. at 746.

68 *Franklin II*, 714 F.2d at 1240.

Franklin II found *Williams* dispositive.[69]

The Third Circuit court's misunderstanding of the *Williams*—*Franklin* difference is apparent from the court's statement:

> The defendant in *Williams* had also argued that a copyrightable work 'must be intelligible to human beings and must be intended as a medium of communication to human beings,' . . . . We reiterate the statement we made in *Williams* when we rejected that argument: '[t]he answer to defendant's contention is in the words of the statute itself.'[70]

Examining the "words of the statute itself," though, leaves no doubt that the court's concern in *Franklin I*, that copyrightabilty be based upon perception, still remains. The "words of the statute itself," as given in *Williams* in the sentence following the one quoted by *Franklin II*, are: "A 'copy' is defined to include a material object in which a work is fixed 'by *any* method now known or later developed, and *from which the work can be perceived*, reproduced, or otherwise communicated, either directly or *with the aid of a machine or device*.' " (emphasis added in *Williams*).[71] But, if a work does not have a human audience, how can it ever be communicated? Certainly, perception by non-humans or mere reproducibility cannot be the basis for copyrightability.

The *Franklin II* court's strongest argument that the computer program's audience is irrelevant with regard to the copyright of object code is that the CONTU report is the law.[72] In 1980, Congress enacted two changes in the Copyright Act that CONTU had recommended. These changes added a new section 117[73] and a new part to section 101.[74] The *Franklin II* court's implicit argument that Congress *also* intended to accept the CONTU majority's position regarding the copyrightability of object code, i.e., that object code is

---

69 The *Franklin II* court said:

> Certain statements by the district court suggest that programs expressed in object code, as distinguished from source code, may not be the proper subject of copyright. We find no basis in the statute for any such concern. Furthermore, our decision in *Williams Electronics, Inc. v. Artic International, Inc.* . . . laid to rest many of the doubts expressed by the district court.

Id. at 1246-47.

70 *Id.* at 1248.

71 *Williams,* 685 F.2d at 877.

72 *Franklin II*, 714 F.2d at 1247-48.

73 The new § 117 concerns limitations on an owner's exclusive rights in computer programs. 17 U.S.C. § 117 (1976), *amended by* 17 U.S.C. § 117 (1980).

74 The 1980 amendment adds the following definition of a computer program: "A 'computer program' is a set of statements to be used directly or indirectly in a computer in order to bring about a certain result." 17 U.S.C. § 101 (1976), *amended by* 17 U.S.C. § 101 (1980).

copyrightable without regard to its audience, is strained. If the term "expression" in section 102 of the Copyright Act is not going to require perceptibility, then a clearer indication of Congressional intent then the 1980 changes in section 117 and section 101 is necessary.

IV. Conclusion

Whether object code can ever be copyrighted depends upon whether object code has an "author" and is a "writing" within the meaning of the copyright clause of the Constitution. Given the broad interpretation of these words by the Supreme Court, it is likely that object code, in any phase, can be copyrighted.

Whether object code is currently copyrightable depends upon whether object code is a "work of authorship" and is an "expression" within the meaning of section 102 of the Copyright Act. Since object code fits within the definition of a "work of authorship," only "expression" presents a problem. If expression requires human perceptibility, which it seems it should, then object code will only be copyrightable if the computer's output is perceptible to one of the five senses. If one is willing to broadly construe "perceptible," then object code is copyrightable. For example, one sees the pages of a book, one tastes and smells a roast cooked in a microwave, one hears a radio with "computerized" tuning, and one feels comfortable in a building with controlled temperature.

Admittedly, perceivable results are not necessarily the direct output of a computer program, but perceivable interaction with human life is the goal desired. Although the machine creates the perceivable results, copyright protection should extend to the program, in source code, object code, or ROM, that causes the machine to create perceivable results, because behind every program there is an author at work, reaching an audience on a scale never before possible.

*James Canfield**

* An earlier version of this note was entered in the 1983 Nathan Burkan Memorial Competition (ASCAP).