UIC John Marshall Journal of Information Technology & Privacy Law

Volume 9
Issue 1 *Computer/Law Journal - Winter 1989*

Article 3

Winter 1989

Microcode - Idea or Expression?, 9 Computer L.J. 61 (1989)

Robert Steinberg

MICROCODE—IDEA OR EXPRESSION?

By ROBERT STEINBERG*

I. INTRODUCTION

> When Intel invented the first microprocessor in 1971, few people realized the implications it held for the electronics industry. From its humble beginnings on the drawing board, the microprocessor soon blossomed into a force that created an entire new market, dramatically altering the landscape of computer science. One look at the present multi-billion dollar microelectronics industry and its bulging roster of companies and concerns . . . support[s] this contention.[1]

Although corporations in the microprocessor industry have expanded greatly in the last fifteen years, technological innovation in this industry has stabilized. Domestic industrial innovators, after spending millions of dollars on research, are now facing competition from foreign corporations that copy and reproduce their technology. By enhancing the microelectronics in the microprocessors and then selling the technology at lower prices, these foreign corporations have captured large shares of the domestic market. Domestic microprocessor manufacturers are in a difficult position, without legal recourse to curtail foreign usurpation of their existing and developing technologies. Whether legal protection for microprocessor technology is promulgated will substantially affect the national economy. Judicial rulings establishing the extent of protection to be afforded microprocessors will determine the microprocessor's long-term availability, price, quality, and form.[2] This

---

 Versions of this Article appear in 27 JURIMETRICS J. 173 (1987) entitled, NEC v. Intel: *The Battle Over Copyright Protection for Microcode*, 13 NEW MATTER, Spring 1988, at 2 entitled, *Copyright Issues Involving Microcode*, and in 2 INT'L COMPUTER L. ADVISOR, Apr. 1988, at 4 entitled, *Microcode Draws the Line on Idea and Expression*.

Robert Steinberg is an associate with Irell & Manella, Los Angeles, California. He was an associate editor to the GEO. J. OF L. & TECH. and Legal Intern to the Hon. Jean Galloway Bissell, U.S. Ct. App. Fred. Cir.; J.D. Georgetown Univ. Law Center; B.S. Systeme Engineering *Magna Cum Laude*, Univ. of Pennsylvania; B.S. Economics *Cum Laude*, Wharton School of Finance.

1. Rant, *Extending the Legacy of Leadership: The 80386 Arrives*, SOLUTIONS, Nov.-Dec. 1985, at 2, 2. Many of these companies were founded during the explosive fourteen years after 1971.

2. NEC Corp. v. Intel Corp., 645 F. Supp. 590 (N.D. Cal. 1986), *vacated*, NEC Corp. v.

Article describes the recent technological and legal developments relating to microprocessor microcode and concludes that limited copyright protection for microcode is merited.

Microprocessors process information by executing sequences of assembly instructions (user-oriented instructions) provided by external software, a human programmer, or a user. For example, the Intel 80386 receives assembly instructions and operating software instructions, which it uses to solve problems. A sequence of these assembly instructions, describing how to perform a certain task, is called an "assembly program." The microprocessor translates each instruction of the assembly program into a lower level, more basic, language called "microcode." Microcode is the industry term for the software inside the microprocessor, consisting of sequences of microinstructions forming microprograms.

Microprograms are rarely more complicated than adding two numbers, checking a resulting number to see if it equals zero, or moving a piece of electronic data from one part of the microprocessor to another. The microprocessor converts each assembly instruction into about four microinstructions. Since a sequence of twenty assembly instructions generates approximately eighty corresponding microinstructions, microprocessor designers attempt to make microinstructions and programs as simple as possible to reduce the complexity and cost of microprocessor electronics. Although the resulting micro-language is very elementary, it would be difficult and tedious for people to use directly. Microprocessors are, therefore, programmed in assembly language, and, through the use of special programs called compilers and interpreters, in high-level languages such as BASIC, FORTRAN, and Pascal.

Microprograms are copyright protectable as long as they are not wholly dictated by the microprocessor's functional considerations and allow programmers to exercise discretion in designing the microprogram expression.[3] Since many microprograms are highly func-

---

United States D. Ct. N.D. Cal., 835 F.2d 1546 (9th Cir. 1988). In this case, NEC, a Japanese microprocessor manufacturer, brought an action to obtain a declaration of invalidity or noninfringement as to copyrights protecting Intel microcode. Intel, a domestic computer manufacturer, claimed that the microcode used in the imported NEC microprocessors was a direct copy of the microcode in the Intel microprocessors. In his pretrial findings of fact and conclusions of law, Judge Ingram concluded copyright protection for microcode was merited. *Id.* at 595. However, Judge Ingram later disqualified himself for having a small financial interest in Intel and ordered his decision to be vacated. Judge Gray (C.D. Cal.) has been appointed to replace Judge Ingram. Trial was set to begin in June 1988; however, as of this printing, no results were yet published.

3. Laurie & Everett, *The Copyrightability of Microcode: Is it Software or Hardware . . . or Both?*, 2 COMPUTER LAW., 1 (1985). *See also* 1 M. NIMMER & D. NIMMER, NIMMER ON COPYRIGHT § 2.01[B], at 2-14.1 (1988).

tional and programmers cannot exercise discretion in their design, the scope of protection for these expressions is limited.

II. COPYRIGHTABILITY OF MICROCODE

The question of whether microcode is copyright protectable raises some of the most fundamental questions in copyright law. Much of the debate involves the interpretation of the language in sections 101, 102(b), and 113(b) of the Copyright Act.[4]

The Copyright Act as amended defines a computer program as "a set of statements or instructions to be used directly or indirectly in a computer in order to bring about a certain result."[5] While this definition has been interpreted to resolve whether high-level computer software programs are works of authorship under the Act,[6] it does not clearly determine whether Congress meant to include all microprocessor-internal microcode programs as copyrightable subject matter.

In section 102(b), the Copyright Act provides that "[i]n no case does copyright protection for an original work of authorship extend to any ideas, procedure, process, system, method of operation, concept, [or] principle . . . ."[7] These undefined terms raise a basic issue regarding when something (*e.g.*, a microprogram or microinstruction) should be considered an uncopyrightable idea instead of a copyrightable expression. Commentators refer to this problem as the "idea-expression dichotomy."[8]

Section 113(b) of the Copyright Act provides that "[t]his title does not afford . . . any greater or lesser rights with respect to the making, distribution, or display of the useful article so portrayed than those afforded to such works under the law, whether title 17 or the common law or statutes of a State . . . ."[9] This section suggests that "copyright in a pictorial, graphic, or sculptural work, portraying a useful article as such, does not extend to the manufacture of the useful article itself."[10] Thus, this section raises another basic question: When is something (*e.g.*, a microprogram or a microinstruction) so elemental in nature that

4. 17 U.S.C. §§ 101-810 (1982) [hereinafter the Copyright Act of 1976].

5. *Id.* at § 101.

6. *See* Apple Computer, Inc. v. Franklin Computer Corp., 714 F.2d 1240, 1248 (3d Cir. 1983), *cert. dismissed*, 464 U.S. 1033 (1984).

7. Copyright Act of 1976, *supra* note 4, at § 102(b).

8. *See, e.g.*, M. NIMMER & D. NIMMER, *supra* note 3, at § 2.03[D] (discussing Copyright Act of 1976, *supra* note 4, at § 102(b)). The term "literary works" includes computer programs to the extent that they incorporate authorship in the programmer's expression of original ideas, as distinguished from the ideas themselves.

9. Copyright Act of 1976, *supra* note 4, at § 113(b) (citations omitted).

10. H.R. REP. NO. 1476, 94th Cong., 2d Sess. 105, *reprinted in* 1976 U.S. CODE CONG. & ADMIN. NEWS 5659, 5720 (quoting the 1961 Report of the Register of Copyrights).

it is the same as the useful article it represents and thus not copyright protectable? This problem is often referred to as the "useful article doctrine."[11]

A. MICROPROGRAMS ARE COMPUTER PROGRAMS AS DEFINED BY 17 U.S.C. SECTION 101

Section 101 of the amended version of the 1976 Copyright Act classifies software as statutory subject matter only if it meets the statutory definition of a computer program, "a set of statements or instructions to be used directly or indirectly in a computer in order to bring about a certain result."[12] In accordance with this definition, each microinstruction can be an "instruction," and the microprogram can be a "set" of instructions. The microprocessor directly uses the microprogram to generate electrical impulses in order to "bring about a certain result," specifically, execution of an assembly instruction.[13]

Microprograms, therefore, have all the characteristics of high-level programs except that they are at the bottommost layer within the machine. Nevertheless, this characteristic is a very important distinguishing feature. At this level, many microprograms are dictated solely by the microprocessor hardware, constraining the programmer's ability to use discretion in their design. In other words, unlike higher level programs, which meet the vague section 102(a) requirement of "authorship," many microcode programs arguably do not meet this requirement because they are engineered.

B. THE IDEA-EXPRESSION DICHOTOMY: MICROINSTRUCTIONS ARE IDEAS AND MANY MICROPROGRAMS ARE NOT EXPRESSIONS

In *Apple Computer, Inc. v. Franklin Computer Corp.*, the court stated that if other programs can be created to perform the same function as Apple's operating system program, the other programs are simply additional expressions of the underlying idea or process and, hence, are copyrightable.[14] Thus, to determine whether a particular microprogram is copyrightable, one must consider the possibility that the logical sequence underlying an expression may be expressed in a number of ways, each requiring a different combination of meaningful

11. *See* Copyright Act of 1976, *supra* note 4, at § 101.

12. *Id.*

13. Laurie & Everett, *supra* note 3, at 6. An individual microinstruction is essentially a one-step computer program consisting of a collection of codes, each code contained in a particular subfield of the microinstruction according to a predefined format. Each code comprises one or more binary values (*i.e.*, one of two voltage levels) which, when loaded into the microinstruction register, generates a set of control signals that bring about a particular result within a processor.

14. 714 F.2d at 1253.

microinstructions.[15] In *Whelan Associates v. Jaslow Dental Laboratory*, the court stated, "[w]here there are various means of achieving the desired purpose, then the particular means chosen is not necessary to the purpose; hence, there is expression, not idea. This test is necessarily difficult to state and it may be difficult to understand in the abstract."[16] Clarifying its position, the court continued:

> We do not mean to imply that the idea or purpose behind *every* utilitarian or functional work will be precisely what it accomplishes, and that structure and organization will therefore always be part of the expression of such works. The idea or purpose behind a utilitarian work may be to accomplish a certain function *in a certain way* . . . and the structure or function of a program might be essential to that task.[17]

Thus, the question is whether another programmer could use his own discretion to create a different expression that performs the same function as the original microprogram, or would he be forced to write the same microprogram because it can only be designed one way.

Under *Whelan*, if a particular logical sequence could be implemented by two different microprograms of different lengths, each microprogram might represent a separate expression of the underlying idea. If the two microprograms both implement and define the same function or process and each contains different meaningful instructions, both might be copyrightable. This situation would be difficult to find in most microprocessors. Many microprograms represent groups of microinstructions whose number and order of sequence is dictated solely by the microprocessor hardware. The designing programmer cannot exercise any discretion in choosing the particular sequence of instructions necessary to perform the desired task. In these situations, the microprograms will not be copyrightable because each program can be represented in only one way.

Arguably, no microinstruction is copyrightable because each is completely dictated by the microprocessor hardware. Indeed, to enhance judicial economy and to maintain consistency in judicial decision making, courts could adopt the bright-line position that every microinstruction, regardless of type, is a non-copyrightable idea. This view of a microinstruction is analogous to the fact that a phrase or word is not copyrightable.[18]

---

15. *Id.* The court made no finding as to whether some or all of Apple's operating system programs might represent the only means of expressing the underlying idea. Apple was able to show that there were other ways to implement the programs copied by Franklin.

16. 797 F.2d 1222, 1236 & n.28 (3d Cir. 1986), *cert. denied*, 107 S. Ct. 877 (1987).

17. *Id.* at 1238, n.34 (citation omitted).

18. *See, e.g.*, Kanover v. Marks, 91 U.S.P.Q. 370 (S.D.N.Y. 1951) (plaintiff's cards and reports adapted for use in servicing consumer electronics were not infringed by defendant's copying and manufacturing a "likeness" of them); Smith v. George E. Muehlebach

By definition, each step (*i.e.*, instruction) in a microprogram must be meaningful, essential, and unchangeable. In theory, if a microprogram of vast length were necessary to define and implement a unitary process or function, that program would not be copyrightable because of the merging of the idea and its expression. Stated differently, if the microprogram expression is at its highest level of abstraction, it probably is not protectable. On the other hand, if the programmer can make choices involving creativity of expression at lower levels of abstraction, then the microprogram is protectable.

C. MANY MICROPROGRAMS ARE UTILITARIAN AND NOT COPYRIGHT PROTECTABLE

At the microcode level, the idea-expression dichotomy and the utilitarian doctrine merge. When there is only one or a few ways of depicting an expression of a microprogram, the expression becomes the very idea it depicts.[19] Such expressions are not copyrightable.[20] Additionally, when there is only one or a very few ways of representing a program, the program becomes the utilitarian and functional object that it represents.[21] Nor are these programs copyrightable.[22]

Many microprograms are "useful articles"[23] and are functional in specific hardware. By definition, microprograms are the industry's best attempt at creating the most efficient software reducing computer processing to the shortest time possible. Unlike higher level programs (*e.g.*, operating systems and application programs), many microprograms do not allow the programmer to exercise discretion in designing the program expression. Many microprograms cannot be redesigned and still function in the same computer hardware.[24] Further-

---

Brewing Co., 140 F. Supp. 729 (W.D. Mo. 1956) (addition of a short set of musical tones simulating the sound of a clock ticking to the words "Tic Toc" was not a copyright protectable work because the addition added nothing of consequence to the phrase, "Time for Muehlebach").

19. Apple Computer, Inc. v. Franklin Computer Corp., 714 F.2d 1240, 1253 (3d Cir. 1983).

20. Copyright Act of 1976, *supra* note 4, at § 102(b).

21. H.R. REP. NO. 1476, *supra* note 10, at 105, U.S. CODE CONG. & ADMIN. NEWS at 5720.

22. Copyright Act of 1976, *supra* note 4, at § 113(b).

23. *See e.g.*, Durham Indus. v. Tomy Corp., 630 F.2d 905 (2d Cir. 1980) (utilitarian features of toys are not copyright protectable); Tayler Instrument Co. v. Fawley Brost Co., 139 F.2d 98 (7th Cir. 1943) (things functioning as machine parts are not copyright protectable due to their utilitarian nature). Because copyright law has traditionally been loath to grant protection to utilitarian objects, it logically follows that courts have also rejected claims of copyright protection for mechanical devices.

24. Specifically, the microprocessor provides the internal time clock for events occurring in a computer; therefore, if any instructions are added to a microprogram the surrounding computer functions may not operate properly due to timing mismatches. When

more, many microprograms are distinguishable from higher level programs in that they are implicit and necessary to their target microprocessor. Accordingly, one can argue that each microprogram is a part of the microprocessor. The computer[25] can operate without any one specific application or operating system program, but it is absolutely necessary that the microprocessor have certain microprograms in order to perform the operations specified by the higher level programs that were discussed in *Franklin*.[26]

If copyright protection is granted to the more basic microprograms, the protection will extend to more than just the microprograms themselves; the machine, or microprocessor, will also be protected. Copyright protection should not be afforded in such cases because it would extend into the realm of patent law.y[27] Allowing protection would enable the microprocessor manufacturers who own copyrights on these programs to also have the exclusive rights to produce microprocessors that have the specific architecture used by the programs. This automatically occurs because only a certain microprocessor architecture can support a particular protected microprogram. Thus, an otherwise unpatentable and uncopyrightable utilitarian design, the specific microprocessor hardware architecture, would be afforded seventy-five years of copyright protection. Although the copyright protection afforded to the hardware architecture would not be nearly as broad as patent protection, the policy of "leaving in the public domain those improvements which do not meet the standard of invention"[28] would still be undermined.

D. PROVING THAT A MICROPROGRAM IS OR IS NOT AN EXPRESSION

The number of ways a program can accomplish a particular func-

---

the software is not absolutely utilitarian and can be redesigned for the machine in which it operates, the programs are copyrightable. For example, computer programs, including both application and operating systems, whether in source or object code, embedded in ROM or expressed in any other medium, are copyright protectable. *See* Apple Computer, Inc. v. Formula Int'l, Inc., 562 F. Supp. 775 (C.D. Cal. 1983), *aff'd*, 725 F.2d 521 (9th Cir. 1984).

25. Microprocessors, mini-computers, and large mainframes are all examples of computers.

26. 714 F.2d at 1240.

27. For an explanation of the relationships between computer programs (and microprograms) and copyright law and patent law, see NATIONAL COMM'N ON NEW TECHNOLOGICAL USES OF COPYRIGHTED WORKS, FINAL REPORT 16-17 (1979) [hereinafter CONTU REPORT].

28. Samuelson, *CONTU Revisited: The Case Against Copyright Protection for Computer Programs in Machine-Readable Form*, DUKE L.J. 663, 736 (1984), (Professor Samuelson raises the issue of copyright/patent overlap with an illuminating example concerning the design of a new airplane wing).

tion is a technical question of fact rather than a question of law. The court would determine a specific microprogram's copyrightability by evaluating the evidence. For example, to show that only a limited number of microprograms can accomplish a specific function, one must use outside experts testifying as to the facts and premises that demonstrate the limited nature of the particular code sequence. Since expert testimony would be the deciding factor in determining whether the individual microprogram is copyrightable, the experts selected must be capable of explaining why the function could not be expressed in different ways. An expert could establish that there is only one way to write a program by testimony showing that the microprocessor architecture constrained the programmer's choice of the particular sequence of microinstructions. In addition, an expert who is able to dissect the opposition's arguments and expose their fallacies would be extremely valuable.

IV. COPYRIGHT INFRINGEMENT OF MICROCODE

Once a court determines which microprocessor microprograms adequately fulfill the requirements for copyright protection, a determination regarding copyright infringement can be made.

A. TRANSLATION OF MICROCODE INTO DIFFERENT MICROCODE

Whelan v. Jaslow dealt with a complex applications program that the defendants converted from a language used on one type of machine (an IBM Series I) to language used on a different machine (an IBM-PC). The court found that, due to the general complexity of the application program, it would not have been difficult to design the application software to follow an entirely different order and sequence. Even though the computer languages of the two systems were entirely different, the sequences of expression—or pattern—in the programs were substantially similar, and the court found infringement.[29]

Where a program is complex and a wide range of expressions with varying order and sequence could implement the same idea, a close reading of *Whelan* suggests a relatively low threshold of finding substantial similarity.[30] However, as discussed above, many microprograms are invariable in their order and sequence and lack complexity. Thus, *Whelan* suggests that there must be a very high degree of similarity in such microprograms for copyright infringement to occur; even a small change from the original microcode expression's order and sequence should be enough to avoid a finding of substantial similarity.

29. 797 F.2d at 1230.
30. *Id.* at 1238-45.

Programmers are allowed, absent a contractual agreement to the contrary, to read original, copyrighted microprograms (assuming they have proper access) and to use the ideas embodied in these programs in preparing their own works.[31] This can be true even though the process amounts to merely translating the original program into a new one. According to the CONTU Report, "the availability of alternative noninfringing expressions is the rule rather than the exception."[32] Furthermore, a line-for-line translation may have to occur in many instances because the original microprogram is really nothing more than the formal logic expression of the underlying idea. In such cases, conversion of the microcode expression into another form amounts only to *translation of an idea* into a new microcode expression and not the impermissible *copying of an expression* into a new microcode expression.[33]

B. REARRANGING THE MICROCODE EXPRESSION TO AVOID INFRINGEMENT

Merely altering the starting addresses of microprograms does not avoid copyright infringement. The starting address is the location of the first microinstruction of a microprogram that implements a given assembly instruction. Changing a microprogram's starting address is no different from changing the starting page number of a chapter in a textbook, since rearranging the chapter's location does not change the sequence and pattern of the book's underlying expression. While there are probably no cases where this has occurred, a court would likely find this reorganization insufficient to avoid copyright infringement under the "pattern test."[34] Similarly, a court would likely find the alteration of microprogram starting addresses affects only the organization of the

31. *See* Continental Casualty Co. v. Beardsley, 253 F.2d 702, 706 (2d Cir. 1958), *cert. denied*, 358 U.S. 816 (1958). *See also* Harcourt, Brace & World, Inc. v. Graphic Controls Corp., 329 F. Supp. 517, 525 (S.D.N.Y. 1971) (cites *Beardsley*).

32. CONTU REPORT, *supra* note 27, at 20 n.106.

33. *See* Copyright Act of 1976, *supra* note 4, at § 102(b) ("In no case does copyright protection for an original work of authorship extend to any *idea, procedure, process, system, method of operation*, concept, principle, or discovery, regardless of the form in which it is described, explained, illustrated, or embodied in such work.") (emphasis added). *See also* H.R. REP. NO. 1476, *supra* note 10, at 57, U.S. CODE CONG. & ADMIN. NEWS at 5670 ("Section 102(b) is intended, among other things, to make clear that the *expression* adopted by the programmer is the copyrightable element in a computer software program, and that the actual *processes* or *methods* embodied in the program are not within the scope of the copyright law.") (emphasis added).

34. *See* Chafee, *Reflections on the Law of Copyright: I*, 45 COLUM. L. REV. 503, 513-14 (1945); *see also* Eisenschiml v. Fawcett Publications, Inc., 246 F.2d 598, 603 (7th Cir. 1951), *cert. denied*, 355 U.S. 907 (1957) (copyright does not protect the abstract idea of the novel or play alone, but it does protect the particular pattern employed in arranging and expressing that idea); Holdredge v. Kruger Publishing Co., 214 F. Supp. 921, 923 (S.D. Cal. 1963).

various microprograms within the microprocessor memory and does not alter a program's actual sequence of microinstructions.

Another practice that does not avoid copyright infringement is rearranging the order of the microinstructions in a microprogram without affecting its function. *SAS Institute, Inc. v. S & H Computer Systems, Inc.*[35] is a recent authority supporting this proposition. Here, the defendants recompiled high-level application software programs to run in a different language. To disguise their direct copying of the software, they edited the source code expression using a "text editor." The text editor allowed the defendant programmers to manipulate the source code expression much as a word processor manipulates text. The programmers were able to select particular lines or groups of lines of code for editing and making desired changes. The court held that this rearrangement of code within a software program did not avoid copyright infringement.

The existing cases on copyright infringement of software, as exemplified by *Whelan* and *SAS*, suggest the following rule: When software program *processing* is unaltered by the rearrangement of memory locations of the component software instructions, it does not avoid copyright infringement. Therefore, courts should also view the mere rearrangement of microinstructions and microprogram starting addresses as an unsuccessful way to avoid copyright infringement.

Applying this reasoning one step further, replacing software with hardware circuitry to implement various steps of a microprogram would probably be a futile attempt at avoiding copyright infringement. A simple alteration of an operation (such as multiplication or division) by exchanging a few lines of microinstructions for hardware circuitry would not change the "sequence of events" (*e.g.*, shifting, looping, etc.) of the original microprogram. Under the "pattern test," the new hardware-supplement ed microprogram would still infringe.[36]

This rule suggests that copyright protection should extend to include the processing of software programs, not just their fixed underlying expressions. However, this extension is problematic because it relies on the programs' similarity when they are being *processed* instead of their similarity when they are fixed in memory. A printout of a program in process may look drastically different from the same program as stored in memory. In addition, when programs are processed, their instructions are no longer within a tangible, fixed medium of expression and thus, they are not subject to copyright protection.[37]

---

35. 605 F. Supp. 816, 826, 829-30 (M.D. Tenn. 1985) (finding of fact no. 65).

36. *See* Sheldon v. Metro-Goldwyn Pictures Corp., 81 F.2d 49 (2d Cir. 1936) (particular sequence of events in a play ruled a copyright infringement).

37. H.R. REP. NO. 1476, *supra* note 10, at 51-53, U.S. CODE CONG. & ADMIN. NEWS at

C. Proving Infringement of a Microcode Program

If an alleged infringer of a microprogram can demonstrate he has independently created the program, he will avoid a finding of copyright infringement. However, this requires the defendant to present a detailed documentary record to chronicle his efforts in developing the microprogram. This particular issue has not yet been dealt with regarding microcode. However, the issue has arisen in cases dealing with the copying of higher level application programs (*Whelan* and *SAS*) and operating systems (*Formula* and *Apple v. Franklin*). The courts in each of these cases found copyright infringement largely because the defendants were unable to produce a paper trail to document the independent creation of the subject matter in question.

Furthermore, in *Whelan* and *E.F. Johnson Co. v. Uniden Corp. of America*,[38] the courts held that finding copyright infringement requires direct evidence of copying and a showing of both access and substantial similarity. Where the plaintiff demonstrates that the defendant had access to the plaintiff's work and the works are substantially similar, a rebuttable presumption exists that the defendant's work is copied. In *Whelan*, the district court relied on expert testimony that the defendant "must have had access to [the plaintiff's Dentalab] source code to be able to understand the system and to be able to use all the same sequential operations."[39] The defendant was not able to rebut this presumption and was found to have infringed on the plaintiff's copyright. Likewise in *SAS*, the defendant was also found to have infringed on the plaintiff's copyright. Again, the court found infringement based on access and substantial similarity since the defendant did not produce sufficient documentation to rebut the resulting presumption of copying.[40]

III. CONCLUSION

Within the limits suggested above, many microcode programs are copyright protectable. The limitations on their scope of protection, however, are fairly substantial. Consequently, many domestic microprocessor manufacturers are looking for other ways to protect their innovations. Patent law, trade secret law, and chip mask all offer potential means of extending legal protection for microcode. Subse-

5664-66 (an unfixed work of authorship, such as an improvisation or unrecorded choreographic work, performance, or broadcast, would not be eligible for federal statutory protection under 17 U.S.C. § 102).

38. 623 F. Supp. 1485, 1493 (D. Minn. 1985).

39. 609 F. Supp. 1307, 1316 (E.D. Penn. 1985) (quoting notes of testimony of Dr. Moore), *aff'd*, 797 F.2d 1222 (3d Cir. 1986).

40. 605 F. Supp. at 816.

quent articles will examine the efficacy of these approaches as possible solutions.