

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

ROKU, INC. and VIZIO, INC.,
Petitioners,

v.

ANCORA TECHNOLOGIES INC.,
Patent Owner.

Case IPR2021-01406
U.S. Patent No. 6,411,941

**DECLARATION OF ANDREW WOLFE, PH.D., IN SUPPORT OF
PETITIONERS' REPLY TO PATENT OWNER'S RESPONSE**

Mail Stop "PATENT BOARD"
Patent Trial and Appeal Board
U.S. Patent & Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450

TABLE OF CONTENTS

I.	INTRODUCTION	1
II.	HELLMAN DISCLOSES THE “AGENT” LIMITATION.....	2
A.	Patent Owner’s Construction Requiring that the Verification Structure Be Set Up Solely by “OS-Level Software” Is Unsupported and Incorrect.....	2
B.	Hellman Renders Obvious the Claimed Agent Under Dr. Martin’s Construction.	9
1.	A POSA Would Have Implemented Hellman in Software Form.	9
2.	A POSA Would Have Implemented Hellman As OS-Level Software, Specifically.....	16
III.	THE COMBINATION OF HELLMAN AND CHOU DISCLOSES THE “VERIFICATION STRUCTURE” LIMITATION.....	24
IV.	A POSA WOULD HAVE BEEN MOTIVATED TO COMBINE HELLMAN WITH CHOU AND SCHNECK.	26
A.	Patent Owner’s Arguments Appear to Be Premised on a Misunderstanding of the Proposed Combination.....	26
B.	A POSA Would Have Been Motivated to Store Hellman’s Value “M” in Hellman-Chou’s Modified BIOS EEPROM.....	31
C.	Storing Hellman’s Encrypted “Authorization A” in “Non-Volatile Memory 37” Would Not Render Hellman Inoperable for its Intended Purpose.	32
V.	CONCLUSION.....	35

I. INTRODUCTION

1. I previously submitted a declaration (EX1003) in this matter on behalf of Petitioners in support of their petition filed on August 24, 2021. I understand that the Board has instituted review, and that Patent Owner has submitted a response. I also understand that Patent Owner's expert witness, Dr. David Martin, has submitted a declaration in support of Patent Owner's response (EX2018). I have been asked to provide my technical review, analysis, and insight regarding both the Patent Owner's response and Dr. Martin's supporting declaration. As explained in more detail below, I disagree with many of Dr. Martin's opinions and analysis.

2. My background and qualifications were provided in paragraphs 6-17 of my prior declaration, and a copy of my CV was appended thereto as Appendix A.

3. Since my prior declaration, I have reviewed and considered the following additional materials:

Exhibit	Description
--	Patent Owner's Response (Paper 22)
1035	Transcript of the Deposition of David Martin, Ph.D., July 14, 2022
1037	Denon DP-35F/DP-45F Instruction Manual, Nippon Columbia Co., Ltd.
1038	Excerpt from <i>Dictionary of Computing</i> , 4 th ed., Oxford University Press, 1996
1039	U.S. Patent No. 5,568,552 to Davis
1040	Guttman, B., et al., <i>Computer Security</i> , National Institute of Standards and Technology, 1995)

Exhibit	Description
1041	Kaliski, B., “PKCS #1: RSA Encryption,” RFC 2313, The Internet Society, Network Working Group, March 1998
1042	U.S. Patent No. 5,724,425 to Chang et al.
1043	U.S. Patent No. 5,935,246 to Benson

II. HELLMAN DISCLOSES THE “AGENT” LIMITATION.

A. Patent Owner’s Construction Requiring that the Verification Structure Be Set Up Solely by “OS-Level Software” Is Unsupported and Incorrect.

4. Dr. Martin contends that the claimed “agent” is limited to pure software running “at the OS level.” EX2018, ¶129; EX1035, 140:18-141:12. As an initial matter, this construction is vague and unclear. Dr. Martin does not explain what “OS-level” means in this context, provide any examples of OS-level programs, or offer guidance about how to determine whether a program operates at the OS level. In his deposition, Dr. Martin seemed to provide varying criteria for making such a determination:

- OS-level software “relates to programs that are running that use the running operating system services, as part of their operation,” EX1035, 100:8-22;
- “OS-level software can be thought of as running through the operating system,” *id.*, 101:19-102:4;
- OS-level software “rel[ies] on operating system services and is doing so after the operating system is running,” *id.*, 102:5-9, 105:4-10;

5. These criteria are themselves ambiguous. Nevertheless, I disagree with Dr. Martin's construction limiting the claimed "agent" to OS-level software based on the criteria set forth above. As an initial matter, the term "agent" is not limited to a pure software implementation. "Agent" is generally understood in the art to encompass both software and hardware. For example, the Oxford Dictionary of Computing (4th ed. 1996) defines "agent" as any "autonomous system that receives information from its environment, processes it, and performs actions on that environment." EX1038, 11. The dictionary goes on to say that agents "may be software, *hardware*, or both." *Id.* (emphasis added). Many patents and articles describe agents in various contexts consistent with this definition. For example, U.S. Patent No. 5,568,552—an Intel patent filed in 1995—describes a "hardware agent" for enforcing software licenses. The hardware agent comprises a processing unit and non-volatile memory that stores encryption keys for determining whether particular software is licensed. *See* EX1039, 1:19-25, 3:1-10, 8:55-9:12. Thus, I disagree that agents are limited to software.

6. But even if the claimed agent in the '941 patent were limited to a software-only implementation, it makes little sense to refer to the agent as if it acts alone, without the assistance from any hardware, to set up the claimed verification structure. All software operates using hardware; software, by itself and in the abstract, is not capable of performing any functions, let alone those claimed in the

'941 patent. Dr. Martin seemed to acknowledge this at his deposition. EX1035, 106:12-107:10, 111:19-22 (acknowledging that the claimed agent runs on a computer or multiple computers).

7. The plain language of the claims supports this commonsense understanding. Claim 1 of the '941 patent recites “using an agent to set up a verification structure” in non-volatile memory. EX1001, 6:64-65; *see also id.*, 8:39-40 (independent claim 18 reciting a similar limitation). The agent is not that which uses the agent. Thus, by its plain terms, the claim merely requires that an agent be “used” to set up the claimed verification structure; it does not foreclose hardware from acting with the agent to do so. Again, Dr. Martin expressly acknowledged this at his deposition. EX1035, 129:9-130:22 (Dr. Martin testifying that while claim 1 “requires the use of the agent to set up the verification structure,” it “does not exclude the possibility of using additional other entities or operations in service of using an agent to set up a verification structure”); *see also id.*, 131:14-19 (“As I read this limitation, I don’t see anything precluding the use of hardware in setting up the verification structure.”).

8. On this point, I note that claim 3, which depends from claim 1, recites that the verification structure is set up using a “license authentication bureau.” EX1001, 7:5-20; *see also id.*, 3:33-44 (noting that the bureau “can participate in ... establishing the license record in the second non-volatile memory”), 3:51-61

(stating that “setting up a verification structure” can be performed “[a]ccording to one example of using the bureau”). The ’941 patent makes clear that the claimed bureau is a “telecommunications accessible processor”—i.e., hardware. *Id.*, 3:33-44; EX1035, 112:7-22, 115:3-18 (Dr. Martin acknowledging that the bureau could be hardware). For claim 3 to make sense, claim 1 must be broad enough to permit hardware to be used in conjunction with the claimed agent to set up the verification structure.

9. Likewise, even if the claimed agent is limited to software, it is not limited to OS-level software, as Dr. Martin contends. The specification does not refer to an agent, let alone one that operates at the OS level, specifically. In fact, the ’941 patent specification does not refer to an operating system at all. EX1035, 133:22-135:3, 139:13-16. And the only passages of the ’941 patent that describe setting up the verification structure do not specify where the agent resides or when it is executed. *See* EX1001, Abstract, 1:59-62, 2:62-3:3, 3:51-61, 6:17-21.

10. Dr. Martin’s cited extrinsic evidence does not change my view. *See* EX2018, ¶126. He relies on various dictionary definitions and articles that use the words “agent,” but none suggests that the described agent is implemented at the OS level. EX2007 is a telecommunications handbook that post-dates the alleged filing date of the ’941 patent and that has a short discussion of “mobile agents.” EX2007, 2-11. That discussion merely refers to an agent as “a program,” but it

does not refer specifically to an OS-level program. *Id.* Similarly, EX2008 and EX2009 define “agent” as “[a] program that performs a background task for a user,” but do not limit the agent to being executed at the OS level. EX2008, 13; EX2009, 18. Finally, EX2010 is an excerpt from a webpage apparently published nearly a decade after the alleged priority date of the ’941 patent and that refers to an agent as “[a] software routine,” without specifying that the agent operates at the OS level. *Id.* Thus, this evidence does not support Dr. Martin’s construction of “agent.”

11. Dr. Martin also relies on the ’941 patent’s file history to support his narrow construction. He suggests that I did “not consider the file history of the ’941 patent” in arriving at my opinions. EX2018, ¶75. But this is incorrect. As I testified at my deposition, I considered “both the original prosecution history and the re-exam history” and concluded that they do not provide a basis to limit the meaning of the term “agent.” EX2026, 23:13-24:1. I still hold that opinion.

12. In my view, the file history establishes that the applicant considered the crux of the alleged invention to be storing the license record in BIOS. In fact, the applicant emphasized that this feature distinguished the purported invention over the prior art. EX1002, 197-201. I see nowhere in the file history where the applicant characterized the agent as an important aspect of the claimed invention, nor where the applicant required the agent to be OS-level software. It seems to me

that the applicant did not in fact believe that an OS-level agent was important, as the specification neither refers to “OS-level” implementation nor an “agent.”

13. The term “agent” was added to the claims to overcome an enablement rejection under 35 U.S.C. § 112. As originally filed, the claims recited “selecting a program” residing in memory and “setting up a verification structure,” without distinguishing between the program being selected and the thing that was setting up the verification structure. *Id.*, 14. The Examiner rejected the claims under § 112 because the “Applicants do not teach the device necessary to edit an EEPROM nor have they made it clear to the Examiner how their system would be implemented in light of the non-trivial processing required to write and erase its data.” *Id.*, 116-117. In response to the § 112 rejection, the applicant replaced the phrase “setting up a verification structure” with “*using an agent to set up a verification structure,*” thereby clarifying that something *other than* the selected program sets up the verification structure. *Id.*, 137; *see also id.*, 127 (summary of Examiner interview stating that “112 corrections” to the claims were discussed); *id.*, 135 (applicant stating that the claims were amended “as agreed upon during the interview”). Thus, it appears that the applicant added “agent” to distinguish the software program to be verified with the software and/or hardware that sets up the verification structure. Neither the plain meaning of “agent,” nor the applicant’s or Examiner’s statements during prosecution limit the agent to OS-level software.

14. As I explained in my opening declaration, Hellman discloses an “agent” in the form of update unit 36, acting alone or with Hellman’s authorization and billing unit 13. EX1003, ¶¶137 (citing EX1004, 9:64-10:13, 10:33-49), 138 (citing EX1004, 6:16-30). In particular, the update unit receives a hash value (H) corresponding to each software package a user is authorized to use and “sets up a structure of memory addresses” (i.e., a table) in the non-volatile memory containing the number of authorized uses (M) for each such hash value. *Id.*, ¶¶135-136 (citing EX1004, 6:16-61, 9:64-10:13). Then, when a user’s request for an additional number of uses (N) of a particular software package is granted by authorization and billing unit 13, the update unit interrogates the memory structure and updates the value M in the particular memory address corresponding to the software package at issue to reflect that the user’s request has been verified. *Id.*, ¶¶137B-138B (citing EX1004, 6:16-7:16, 9:64-10:13). In setting up a memory structure defined by hash values for the purpose of verifying the user’s access to software packages, Hellman’s update unit—again, acting alone or in conjunction with the authorization and billing unit—“sets up a verification structure,” as required by the ’941 patent claims.

15. As I also explained in my opening declaration, Hellman’s update unit, as well as its authorization and billing unit, “would have been implemented by a software routine, potentially along with a hardware module.” *Id.*, ¶137; *see also*

id., ¶¶137A-138B. As explained above, the claims do not exclude an agent acting in concert with hardware—nor could they since software necessarily runs on and interacts with hardware. *Supra* ¶¶6-7. Thus, the mere fact that Hellman’s update unit (and authorization and billing unit) could interact with hardware when carrying out the claimed verification setup step does not undermine my obviousness conclusion.

B. Hellman Renders Obvious the Claimed Agent Under Dr. Martin’s Construction.

16. Additionally, as I explain below, Hellman renders obvious the claimed agent under the criteria that Dr. Martin sets forth as defining OS-level software.

1. A POSA Would Have Implemented Hellman in Software Form.

17. As explained above, a POSA would have recognized that Hellman’s update unit 36—again, alone or conjunction with the authorization and billing unit 16—“would have been implemented by a *software routine*, potentially along with a hardware module.” EX1003, ¶137 (emphasis added). As I noted in my opening declaration, implementing Hellman in software form would confer many benefits, including the ability to change the implementation logic over time, without having to physically disassemble, modify, and reassemble the units. *Id.*, ¶¶137-138. I

reiterated these points throughout my declaration and during my deposition.

EX1003, ¶¶137B, 138B; EX2026, 34:17-19, 35:9-18.

18. I do not read Dr. Martin’s declaration as disputing any of this. Rather, Dr. Martin offers several arguments as to why, notwithstanding the aforementioned benefits, a POSA would not *modify* Hellman to be implemented in pure software form. But Dr. Martin appears to misread my opening declaration. In my view, Hellman would not need to be “modified” at all. Rather, Hellman’s disclosed verification process is in and of itself, and without modification, suitable for software implementation. Again, I emphasized that point repeatedly in my opening declaration. EX1003, ¶¶137 (“[A] POSA would have recognized that the update unit 36 would have been implemented by a software routine, potentially along with a hardware module.”), 137A (“[A] POSA would have understood that it was up to the discretion of the implementer whether to use software, hardware, or a combination of the two.”), 137B (noting that the functions performed by the update unit “were of a type that could be performed in software”); *see also id.*, ¶¶138A-138B.

19. Regardless, I disagree with each of Dr. Martin’s arguments as to why Hellman’s solution allegedly would not be implemented as software. First, Dr. Martin contends that Hellman’s disclosure of certain hardware elements such as “wires, switches, and glue” means that Hellman’s update unit and billing and

authorization unit are limited to hardware. EX2018, ¶¶164-166. I disagree. Every computer system has hardware, but that does not mean that all such systems lack software. Hellman's base unit 12, which is a computer system that includes Hellman's update unit 36, is no different. EX1004, Abstract, 1:8-16, 2:24-29, FIG. 8; EX1035, 145:5-20. The mere fact that Hellman's computer uses "wires, switches, and glue"—as do most computers—does not preclude Hellman's system from also using software to control or manage operations of the update unit (or authorization and billing unit). EX1035, 163:18-164:21 (Dr. Martin acknowledging that software could perform the EEPROM read and write functions described in Hellman).

20. In support of his contentions, Dr. Martin seems to focus on Hellman's disclosure of using its system in the context of the "record industry" and "record players." EX2018, ¶¶170-171. According to Dr. Martin, Hellman's reference to record players "would not bring to mind a pure software implementation." *Id.*, ¶171. As an initial matter, Dr. Martin's suggestion that record players cannot be programmable and do not have software is incorrect. *See* EX1037, 1 (describing record player from the 1980s that I owned as having a microprocessor, which would have run software). In any event, Dr. Martin ignores other portions of Hellman's disclosure—including in the very passage to which Dr. Martin cites—stating that Hellman's system can be used to control software distribution in

“computers,” generally. *See, e.g.*, EX1004, Abstract, 1:8-16, 2:24-29, FIG. 8; EX2018 ¶170 (Dr. Martin acknowledging that Hellman “describes the base unit as a ‘computer’”); EX1035, 145:5-20, 157:19-159:1 (Dr. Martin acknowledging at his deposition that Hellman’s system is not limited to record players). Computers indisputably bring to mind a software implementation.

21. Second, Dr. Martin seems to suggest that there are no processors in Hellman’s system on which verifying software could run. EX2018, ¶164. I disagree. As I made clear during my deposition, Hellman’s base unit—like any computer system—contains at least one processor that could run software. EX2026, 34:12-35:6 (noting that Hellman’s components would be “running on a computer system, which most likely would include a microprocessor”); EX1035, 145:5-20 (Dr. Martin acknowledging that Hellman’s base unit could be a computer, and that computers in 1998 had processors). For example, one such processor would be Hellman’s software player 42, which Hellman describes as “a microprocessor or central processing unit (CPU).” EX1004, 10:66-11:3; EX2026, 34:6-11. Patent Owner argues that a software agent cannot run on Hellman’s software player 42 because such a configuration would be inoperative since the agent “could not run before the software player 42 is ‘activated.’” POR 60 (citing EX1004, 10:44-54)). I disagree. The processor within the software player 42 could be used to run an agent before the player is activated—i.e., before it plays the

software package subject to the license. In any event, as noted above and explained in my deposition, Hellman's software need not run on the software player. Rather, a POSA would understand that it could run on any number of other processors in Hellman's system. EX2026, 31:23-25, 34:12-35:6, 35:4-6.

22. Third, Dr. Martin suggests that implementing Hellman in pure software form would lead to security risks that would offset the many benefits described above. In particular, Dr. Martin states that a software agent in Hellman's system would need to "isolate and protect itself from the very software packages whose authorization is at issue." EX2018, ¶174; *see also id.*, ¶¶164, 175. I disagree that this risk is alone sufficient to dissuade a POSA from implementing Hellman in software form, particularly in view of the benefits described above and in my opening declaration. *Supra* ¶17.

23. In any event, presumably the very same risk that Dr. Martin describes would also apply to a pure software agent in the '941 patent's system. And yet the '941 patent does not describe any way to guard against such risks. Accordingly, it seems that *any* standard mechanism for protecting the agent software could be used. And there were many such mechanisms known in the art at the time of the alleged invention. Indeed, implementing security mechanisms in software form was commonplace at the time of the alleged invention of the '941 patent. *See, e.g.*, EX1040, 45, 55, 157, 182, and Chapter 19 (NIST's Handbook on computer

security describing established mechanisms and “computer security programs” for combatting security threats). By 1998, there were numerous software-based mechanisms for ensuring that a particular piece of software was legitimate and not tampered with. One such mechanism was to use a digital signature or a digital security certificate to ensure that the software to be executed on a computer was identical to the original software provided by the software manufacturer. *See, e.g.*, EX1041 (Network Working Group’s RFC 2313, published in March 1998, describing RSA protocol for using encryption, digital signatures, and certificates to protect data). Numerous computer systems including those used for banking, cellular telephony, and military communications implemented secure software components. *See, e.g.*, EX1042, 1:51-2:7, 3:1-36 (describing various “encryption techniques for enhancing software security” to protect against “viruses,” “worms,” and “[s]ecurity breaches”); EX1043, Abstract, 3:34-52 (describing encryption “protection mechanism for protecting software against copying”). In my view, a POSA would have known to use any of these mechanisms to guard against the risk that Dr. Martin identifies.

24. Additionally, I note that Patent Owner mischaracterizes my testimony to suggest that I opined as to what a POSA could do rather than what a POSA would do based on Hellman’s disclosure. *See, e.g.*, POR, 59-61. But I unambiguously stated in my opening declaration that a POSA would have

recognized that the update unit 36 “*would* have been implemented by a software routine” EX1003, ¶137 (emphasis added); *see also id.*, ¶¶137A (“A POSA *would* have been motivated to implement the update unit 36 in software”), 138-138B (similar testimony vis-à-vis authorization and billing unit 13).

25. I also noted that Hellman “does not explicitly say” whether its update unit should be implemented as software. *Id.*, ¶137A. That observation does not indicate doubt as to the desirability of using Hellman’s software implementation—rather, it confirms that Hellman itself does not teach away or dissuade a POSA from implementing its system in software form. This is particularly true in view of the benefits described above and in my opening declaration. *See supra* ¶17.

26. Finally, as I explained above and as Dr. Martin agrees, even if the claimed agent is limited to software, nothing in the claims precludes the software from working with hardware to set up the claimed verification structure. *Supra* ¶¶6-8. And, indeed, Dr. Martin’s analysis seems to come down to a line-drawing exercise. *See* EX1035, 122:12-123:10 (Dr. Martin acknowledging that the software within a “mixed software/hardware entity” could comprise the claimed agent).

2. A POSA Would Have Implemented Hellman As OS-Level Software, Specifically.

27. As explained above, Dr. Martin articulated varying criteria that he alleges defines OS-level software:

- “relates to programs that are running that use the running operating system services, as part of their operation,” EX1035, 100:8-22;
- “can be thought of as running through the operating system,” *id.*, 101:19-102:4;
- “rel[ies] on operating system services and is doing so after the operating system is running,” *id.*, 102:5-9, 105:4-10;

In my view, a POSA would also have implemented Hellman’s system as *OS-level* software, at least under one or more of the above criteria.

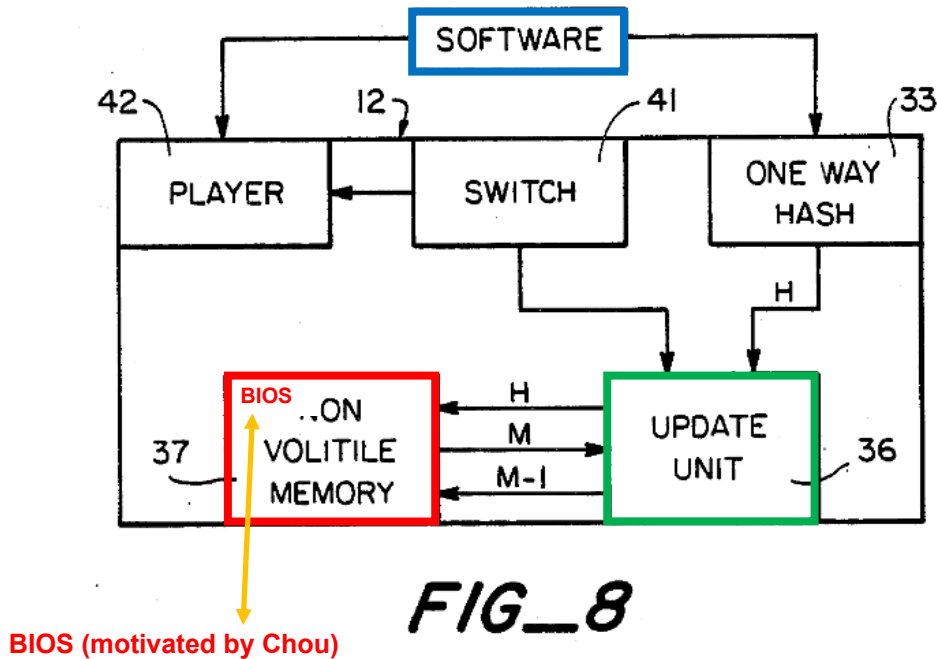
28. As I explained above, a POSA would have understood that Hellman’s system uses a computer as the base unit (which includes the update unit). *Supra* ¶19. And it was well understood at the time of the alleged invention that computers used operating systems. Indeed, I made clear during my deposition that “[a] general purpose desktop computer, like an ordinary PC, would usually have an operating system.” EX2026, 34:1-2; *see also id.*, 31:21-32:23 (noting that Hellman “talks about a computer,” and that a POSA “would assume that a computer has an operating system”), 33:16-18 (“If you bought a computer, a desktop computer, for home use, that would almost always have an operating system.”).

29. I note that Dr. Martin himself acknowledges that operating systems—including, DOS, which was available in the early 1980s—were known at the time of the alleged invention of the '941 patent:

Examples of operating systems that were also known at the time of the invention included Windows NT, Windows 95, DOS, Macintosh OS, and UNIX-based operating systems. (Ex. 2014 [Misra] at 5:67–6:5; Ex. 2019 [Ginter] at 81:35-36.) Each of these commercially available operating systems relies on a component that begins to run automatically when the computer is turned on to start the operating system (often referred to “booting” the computer).

EX2018, ¶73; *see also* EX2026, 32:12-14; EX1035, 99:17-100:1, 109:9-17 (Dr. Martin “agree[ing] that operating systems generally were well known to people of ordinary skill in the art in 1998” and that, “in 1998, it was known that programs could run that would rely on operating system services”). Thus, Hellman’s system is already implementable as an OS-based system.

30. Further, Hellman’s architecture strongly supports the notion that a POSA would in fact have implemented update unit 36 as OS-level software, at least under Dr. Martin’s interpretation of that term. As Figure 8 below shows, Hellman’s architecture includes a software package 17, which uses operating system resources.



EX1004, FIG. 8 (annotated).

In particular, the software is played using software player 42, which is part of base unit 12. The presence of software player 42 in base unit 12 suggests that the base unit is in fact running an operating system. EX1035, 98:4-99:7, 102:20-103:8, 105:11-106:11 (Dr. Martin using similar rationale to justify the existence of OS-level software in the '941 patent's system).

31. Update unit 36 is also part of base unit 12. *Id.*, 144:1-6. The update unit provides access to non-volatile memory 37 (and the BIOS motivated by Chou therein). In particular, update unit 36 allows the system to verify and modify certain M values depending on whether the software package is licensed. *See supra* ¶14. And, in order to perform that verification, the update unit must update M values by performing basic arithmetic, thereby using system resources (e.g., the

operating system's arithmetic control unit, clock, processing power, etc.). Thus, in an OS-based system, because the update unit uses system resources to verify data in non-volatile memory for the purpose of making software available to a user, the agent would be understood to be operating at the OS level under Dr. Martin's criteria. Indeed, in that circumstance, not only would Hellman's update unit "relate to" programs (software 17 and player 42) that use operating system services, the update unit would itself use operating system services and run "through" the operating system. Thus, in that circumstance, Hellman's update unit would satisfy Dr. Martin's criteria for OS-level software.

32. Dr. Martin's own reliance on two passages of the '941 patent in paragraph 128 of his declaration demonstrates that Hellman renders obvious an OS-level agent under Dr. Martin's criteria. EX1035, 135:5-9, 137:16-138:17. The first cited passage is column 1, line 65, to column 2, line 9. That passage explains that the alleged invention (in Patent Owner's view, the purported "agent") uses EEPROM manipulation commands to store, add, and modify items in non-volatile memory of the BIOS:

The resulting encrypted license record is stored in another (second) non-volatile section of the BIOS, e.g. E²PROM (or the ROM). It should be noted that unlike the first non-volatile section, the data in the second non-volatile memory may optionally be erased or modified (using E²PROM manipulation commands), so as to enable to add, modify or

remove licenses. The actual format of the license may include a string of terms that correspond to a license registration entry (e.g. lookup table entry or entries) at a license registration bureau (which will be further described as part of the preferred embodiment of the present invention).

EX1001, 1:65-2:9. Dr. Martin explained at his deposition that this passage “fore shadow[s]” *[sic]* an OS-level agent. EX1035, 138:7-17.

33. Hellman works precisely the same way. As I explained in my opening declaration, Hellman’s “update unit 36 retrieves a value stored at a location in EEPROM, performs integer addition and/or subtraction, and transmits a value to be stored at a location in EEPROM.” EX1003, ¶137B (citing EX1004, 9:64-10:13); EX1035, 148:6-22, 152:8-153:22, 156:7-157:1, 163:18-164:2 (Dr. Martin agreeing that Hellman’s update unit “reads and writes data to its non-volatile memory”). Thus, to the extent the ’941 patent suggests an OS-level agent, so does Hellman.

34. Dr. Martin’s second cited passage of the ’941 patent is column 6, lines 18-28, which likewise says that setting up the verification structure requires accessing non-volatile memory:

Setting up (18) the verification structure includes the steps of: establishing or certifying the existence of a pseudo-unique key in the first non-volatile memory area; and establishing at least one license-record location in the first or the second nonvolatile memory area. Establishing a license-record includes the steps of: forming a license-record by encrypting of the contents used to form a license-record with

other predetermined data contents, using the key; and establishing the encrypted license-record in one of the at least one established license-record locations (e.g. 10-12 in FIG. 1).

EX1001, 6:18-28. Again, as I have already explained, Hellman sets up its verification structure in much the same way. *See supra* ¶33; EX1003, ¶¶133-150 (citing EX1004, 1:39-2:54, 5:57-7:16, 9:64-10:13, 10:33-49, 10:55-65).

35. Thus, to the extent that EEPROM manipulation commands and other memory operations are hallmarks of operations conducted at the OS level in the '941 patent, those operations would similarly be performed at the OS level in Hellman as well.

36. I note that Dr. Martin relies on additional evidence that further supports this conclusion. In paragraph 88 of his declaration, Dr. Martin refers to a white paper that the applicant cited during prosecution of the '941 patent as allegedly describing “a specific embodiment of the invention.” EX1018, ¶88 (quoting EX2011, 129). As an initial matter, the paper states on its face that it is “Confidential,” and it does not appear in the version of the file history available on the Patent Office’s website (EX1002). Even still, the paper supports the notion that OS-level agents were well known in the art. In particular, the paper refers to a well-known command function (referred to as DMI Function 52h) for modifying EEPROM/BIOS:

Beeble will attempt to write license data by utilizing the DMI Function 52h², which is designed to write DMI structures to EEPROM. If BIOS manufactures do not support DMI function 52h, then Beeble will write to a EEPROM generically. Beeble's File System incorporates a driver that contains a library of different EEPROM chips and proper instructions set to write to different chip manufacturers. If the Beeble driver does not recognize the EEPROM, the Beeble driver will attempt to download a new driver from the Beeble License server.

EX2011, ANCC000184 (emphasis added). Dr. Martin acknowledged that DMI Function 52h provides "the ability to manipulate EEPROM memory." EX1035, 174:20-175:4.

37. The quoted passage, in turn, cites to "SMBIOS Reference Specifications" published in March 1998, months before the alleged priority date of the '941 patent. EX2011, ANCC000184 n.2. Thus, as Dr. Martin acknowledges, functions for writing to EEPROM/BIOS were clearly known in the art. EX1035, 178:21-179:14. Further, Dr. Martin testified that the software described in the white paper as using such functions is OS-level software. EX1035, 175:5-176:9.

38. Again, Hellman also writes to its EEPROM, as explained above and in my opening declaration. *Supra* ¶¶33-34. Thus, a POSA would understand that Hellman performs the same functions, which Patent Owner and Dr. Martin seem to suggest comprise one of the hallmarks of an OS-level program.

39. Finally, Dr. Martin drew a temporal and logical distinction between BIOS-level programs and OS-level programs. In particular, he suggested at his deposition that whereas the BIOS runs during startup of a computer, OS-level software runs after startup has completed:

[O]ne of the BIOS's operations, one of its essential start-up operations, is to start the operating system.

And so there is, in that sense, a moment of transition when the operating system first starts running, its services become available, and then an application could rely on those services. When they do so, they're relying on the OS level services, not the BIOS configuration utility that may launch when the computer is first turned on, for example.

EX1035, 103:9-104:2; *see also id.*, 94:11-95:19. Relatedly, Dr. Martin stated in his declaration that the claimed agent “would be understood as a software program or routine *separate from the BIOS.*” EX2018, ¶131 (emphasis added).

40. Under that logic, a POSA would recognize that Hellman's update unit 36 must be implementable as OS-level software because it must be capable of updating data in non-volatile memory *after* startup (and not just during startup). Likewise, Hellman's update unit is “separate from” the BIOS—indeed, under Dr. Martin's and Patent Owner's logic, Hellman does not have a BIOS at all. Thus, the update unit software cannot itself be part of the BIOS.

41. Accordingly, in my view, Hellman renders obvious the claimed agent under Dr. Martin's narrow construction.

III. THE COMBINATION OF HELLMAN AND CHOU DISCLOSES THE “VERIFICATION STRUCTURE” LIMITATION.

42. Patent Owner also argues that Petitioners rely on the same elements of Hellman for disclosing both the “memory of the BIOS” limitation and the “verification structure” limitation of claim 1. POR, 62. Notably, Dr. Martin does not offer any supporting testimony.

43. I disagree with Patent Owner’s attorney argument. As I explained in my opening declaration, the claimed “memory of the BIOS” corresponds to Hellman’s non-volatile memory 37—which Hellman states can be EEPROM—including BIOS as motivated by Chou. EX1003, ¶¶104-106. A POSA would have found it obvious to modify Hellman’s non-volatile memory to include BIOS for all the reasons I explained in my declaration (and below). *Id.*, ¶107.

44. By contrast, as I also explained in my opening declaration, the claimed verification structure corresponds to Hellman’s memory structure within its non-volatile memory 37 (EEPROM), which stores at least one value M at memory addresses defined by at least one hash value H. *Id.*, ¶135. That memory structure is used to verify whether operation of the selected software package 17 is permitted. *Id.*, ¶136. As I explained in my deposition, Hellman’s memory structure could be a data “table” that uses H values as an index. EX2026, 30:1-22. An example of such a table is shown below:

Memory Address	M Value
Address Defined by (H1)	M1
Address Defined by (H2)	M2
Address Defined by (H3)	M3

45. But this memory structure is different than the memory in which it resides. In my view, Patent Owner errs in conflating the two.

46. To the extent Patent Owner suggests that the claimed verification structure must be something other than a data structure within memory, I disagree. Nothing in the '941 patent supports such an interpretation of the claims, and a POSA would understand that a verification structure is merely a data structure (e.g., a data table) used to verify a license. In fact, the '941 patent itself describes its verification structure as a “database,” EX1001, 2:20-26, which is a type of data structure much like Hellman’s memory structure.

47. Patent Owner also suggests that Hellman’s memory structure comprises a “nebulous collection” of hash values. POR, 63. But this suggestion appears to be misplaced. As an initial matter, Hellman’s memory structure stores M values, not H values. Regardless, the important point here is that Hellman’s memory structure uses memory addresses *defined by* the H values. EX1003, ¶135. Further, the M values within Hellman’s memory structure do not comprise a “nebulous collection” of data at all. Rather, as explained above, they are organized in a well-defined table (i.e., a data structure). *See supra* ¶14. The '941 patent claims do not require anything more.

**IV. A POSA WOULD HAVE BEEN MOTIVATED TO COMBINE
HELLMAN WITH CHOU AND SCHNECK.**

**A. Patent Owner's Arguments Appear to Be Premised on a
Misunderstanding of the Proposed Combination.**

48. Patent Owner argues that Petitioners have not shown why a POSA would have been motivated to use “Chou’s BIOS memory in place of Hellman’s non-volatile memory 37.” POR, 52. This is not the proper characterization of the proposed combination. I preface my entire discussion of the Hellman-Chou combination with this opener: “When adding BIOS to the base unit 12 of Hellman, a POSA would have found it obvious to use the non-volatile memory 37 of Hellman to store the BIOS routines.” EX1003, ¶112. Accordingly, the reasons for the combination are based on using Hellman’s existing memory component to store both license record information BIOS information (not replacing Hellman’s memory with Chou’s), with Chou’s BIOS memory serving as the blueprint for this implementation.

49. First, while Hellman does not explicitly teach BIOS, Chou explains that it would be advantageous to use non-volatile memory, such as Hellman’s EEPROM, to store BIOS, and to form a “BIOS memory” that includes both BIOS routines and other information such as passwords. Hellman’s license record is an example of this other information that could be stored BIOS memory with BIOS routines. Dr. Martin alleges that Hellman’s omission of BIOS or BIOS memory is

intentional because the only memory disclosed by Hellman is “normally ‘inaccessible’” to users. EX2018, ¶187. But Dr. Martin’s motivation gives short shrift to Chou’s explicit disclosures regarding how an EEPROM may be implemented. In particular, as I explained in my declaration, Chou explicitly describes implementing an EEPROM as BIOS memory and storing security information, other than the BIOS, in the same BIOS memory. EX1003, ¶115.

50. Next, there are a limited number of choices where to store BIOS in Hellman’s system. Patent Owner does not seem to dispute that all computers have BIOS, and so Hellman’s system must have BIOS even though it is not explicitly disclosed. Chou provides a suggestion and motivation as to which one of Hellman’s memory components to use.

51. I understand that Patent Owner has argued that my testimony is inconsistent with statements in my declaration regarding this point. I disagree with Patent Owner. Starting with my declaration, I stated that “[a]s of May 1998, a typical computer would have had less than five such EEPROM modules, and perhaps only one.” *Id.*, ¶114. I made this statement in the context of Hellman’s disclosure of a non-volatile memory 37 (EEPROM) and other memory modules, such as permanent memory 31 and temporary memory 28, and no “other EEPROM modules.” *Id.* Hellman’s configuration was not unusual as of May 1998. I observed that, in the time period of 1998, a typical computer would have had less than five

EEPROM modules, and even one module. So I concluded that it would have been easy to select Hellman's non-volatile memory 37, from the limited number of choices in Hellman's system, as the location for storing BIOS routines.

52. Turning to my testimony, I was asked whether it was "typical to have more than one EEPROM" in a computer. I responded that this would depend on the computer. A computer put together with components from multiple suppliers could have multiple EEPROMs. I attempted to contrast this with a computer bought off the shelf, such as from Best Buy, where components were preselected. It appears that I misspoke during my testimony and said, in such situations, "it would be not at all common for there to be only one [EEPROM module]." EX2026, 40:12-15. I meant to say that it would be common in those circumstances for there to be only one EEPROM; this is clear from my attempt to contrast computers from multiple suppliers and computers from a single supplier, like Compaq.

53. This testimony is not inconsistent with my declaration where I opined on Hellman's computer. While I testified that it was certainly possible for computers to have more than one EEPROM module, nothing in my testimony contradicts my observations of Hellman as a computer with a single EEPROM module.

54. I also understand that Patent Owner argues it was known that BIOS routines and configuration data occupied more memory than was available in “standard BIOS memory space” and cites column 2, lines 6-12, of a reference, Ewertz, which is reproduced in full, below:

Also, because of new technologies and capabilities such as Extended Industry Standard Architecture (EISA) systems, flash memory and multi-language support for international operation of a computer system, it is becoming increasingly unfeasible to fit all desired BIOS features within the 128K boundary of the IBM PC AT architecture.
EX2015, 2:6-12.

55. Patent Owner’s statements here do not address the proposed combination of using EEPROM to store BIOS. The cited portion of Ewertz does not criticize or otherwise discourage using EEPROM memory to store BIOS and other information as clearly taught by Chou.

56. Turning to the remaining reasons, the third reason is that storing BIOS in Hellman’s EEPROM, which already stores the licensed uses for software, would provide security benefits to Hellman’s system. I understand that Patent Owner argues that Chou relates to disabling the bootup process and does not support a conclusion that storing information in the BIOS would prevent tampering. POR,
53. First, Hellman not Chou, discusses the goal of discouraging tampering.
EX1004, 4:22-26. Chou’s disclosure of using a BIOS for storing sensitive

information merely furthers Hellman's goal of discouraging tampering. As I explained in my declaration, Chou teaches that any attempt to delete the protections to access the sensitive information would result in disabling the bootup process of the computer. EX1003, ¶115. Disabling the bootup process of a computer prevents the computer from starting, which a POSA would understand to be an attempt to prevent tampering with the computer. Accordingly, modifying Hellman's non-volatile memory 37 to include BIOS routines would discourage attempts to access other sensitive information located in that same memory.

57. And finally, it is more efficient to use a component that already exists in Hellman's system rather than adding a new memory module to store the BIOS. I understand that Patent Owner appears to allege that my testimony undercuts this reasoning. Patent Owner first points to my testimony about "lots of ordinary desktop computers that did have multiple [EEPROM] modules." EX2026, 40:23-24. But this testimony was about computers, generally, and unrelated to what is actually described in Hellman's system. So, while it is true that many desktop computers could have multiple EEPROM modules, Hellman describes a computer that includes only one. As I explained above, computers with one EEPROM or multiple EEPROMs were available in 1998. Hellman merely describes one that falls into the former category.

58. Patent Owner also points to my testimony that “[t]here would have to be some hardware involved somewhere” as evidence that some hardware would have to be *added* to implement switch 41 in Hellman’s system. POR, 54. This is not my testimony. My full statement was that “[t]here would have to be some hardware involved somewhere, but the functionality could be turning on an enabled bit in a microprocessor.” EX2026, 35:16-18. That is, the switch 41 could be implemented using existing hardware in Hellman’s system, such as via a microprocessor. Nowhere in my testimony or declaration do I state that hardware would need to be added to Hellman’s system.

B. A POSA Would Have Been Motivated to Store Hellman’s Value “M” in Hellman-Chou’s Modified BIOS EEPROM.

59. I understand that Patent Owner argues that the Hellman-Chou combination lacks a motivation for “storing the value ‘M’ in the memory area that is ‘used by the BIOS,’ as the ‘memory of the BIOS’ term requires.” POR, 55. Ancora then alleges that the combination is premised on assuming the entirety of Chou’s EEPROM chip constitutes “memory of the BIOS.” Patent Owner appears to be mischaracterizing the combination.

60. I explained in my declaration (and above) that the combination is “us[ing] the non-volatile memory 37 of Hellman to store the BIOS routines.” EX1003, ¶112. Hellman’s combined BIOS EEPROM, which stores Hellman’s licensed record “M” and the BIOS, is exactly what is contemplated by the ’941

patent. Figure 1 of the '941 patent shows how licensed records are similarly stored within a non-volatile memory. *See* EX1001, FIG. 1.

61. This combination does not rely on any assumptions, but on Chou's explicit disclosures. Chou describes both BIOS and passwords within the same BIOS memory. EX1005, FIG. 7, 7:14-27; *see also id.*, 3:52-55 ("BIOS memory 15 [] may be a flash EEPROM containing the various executable BIOS routines *as well as* routines for implementing a security function" (emphasis added)). As shown in the above figures, Hellman's non-volatile memory 37 would be implemented to store licensed uses "M+N" as well as BIOS routines in a similar manner as Chou discloses storing BIOS and passwords in his BIOS memory. Hellman's non-volatile memory 37 implemented with Chou's BIOS would therefore constitute the claimed "memory of the BIOS." Moreover, Chou teaches storing both BIOS routines and sensitive information in the same memory, which would have led one of ordinary skill in the art to have implemented Hellman's non-volatile memory 37 to store both Hellman's licensed uses and BIOS routines.

C. Storing Hellman's Encrypted "Authorization A" in "Non-Volatile Memory 37" Would Not Render Hellman Inoperable for its Intended Purpose.

62. I understand that Patent Owner argues that using Hellman's authorization A as the encrypted license record "would render Hellman's system inoperative." POR, 64. Patent Owner argues that "increment[ing] or

decrement[ing]” the value of authorization A is not possible which would make “it impossible for the update unit to change the number of remaining uses.” POR, 65.

Dr. Martin makes a similar argument: that the proposed Hellman-Chou combination is improper because it would be unable enforce the “number of uses” constraint by “decreasing the number of remaining uses when the software is used.” EX2018, ¶201. But Patent Owner and Dr. Martin appear to ignore that the Petition relied on Hellman’s “unlimited number of uses” embodiment. In such an embodiment, the license record may be set to any value that “represent[s] infinity” and Hellman’s update unit 36 would be configured to “not change [this value] when the software package was used.” EX1004, 10:55-65; *see also* Pet., 45 (“M would not need to be incremented or decremented”); EX1003, ¶148 (“A POSA would have recognized that storing authorization A in non-volatile memory 37 could have been especially beneficial where the special value M for an unlimited number of uses was used” and “[w]ith the unlimited uses value, the value M never needs to be incremented or decremented”).

63. Patent Owner also argues that authorization A does not include value M or any other information regarding the number of remaining uses but does not dispute that authorization A includes “N,” representing the number of requested uses. POR, 65. In a scenario where software is being used for the first time, a POSA would understand that the value M (the initial license record) is set to zero

(i.e., $M=0$). EX1003, ¶170. When the requested number of uses N is added to M , the resulting value of M is equal to N (i.e., $0+N=N$). *Id.* Accordingly, when authorization A is stored in memory, N is stored as the license record for the number of uses. *Id.* Dr. Martin disputes this conclusion, arguing that Hellman is not limited only to this scenario and must be able to “acquire additional uses no matter how many are currently permitted.” EX2018, ¶203. But again, in the unlimited-uses embodiment, the request for additional uses only needs to occur one time since the number of uses is never incremented or decremented.

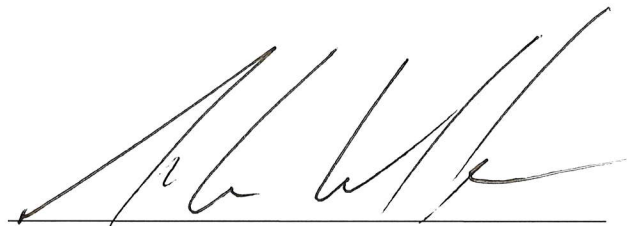
64. Moreover, in Hellman’s “unlimited number of uses” embodiment, I explained that a person of ordinary skill in the art would use authorization A as the license record because Hellman teaches that encrypting authorization A and storing an encrypted value for the number of uses would improve the security of Hellman’s system. EX1003, ¶¶139-146. I pointed out that Schneck described the flaw of storing authorization information in an unencrypted format, such as Hellman’s value M . *Id.*, ¶145. A POSA would have found it obvious to store Hellman’s authorization A in place of value M when considering how to store M in an encrypted form based on Schneck’s disclosure. *Id.*, ¶¶146-147. Dr. Martin does not appear to dispute this conclusion.

V. CONCLUSION

65. In signing this declaration, I recognize that the declaration will be filed as evidence in a contested case before the Patent Trial and Appeal Board of the United States Patent and Trademark Office. I also recognize that I may be subject to cross-examination in the case and that cross-examination will take place within the United States. If cross-examination is required, I will appear for cross-examination within the United States during the time allotted.

I hereby declare that all statements made herein are of my own knowledge and are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code.

Executed on July 22, 2022, in Santa Clara, CA.

A handwritten signature in black ink, appearing to read 'Andrew Wolfe', is written over a horizontal line.

Andrew Wolfe, Ph.D.