

EXHIBIT B

Ancora's Exhibit 2004
HTC v. Ancora IPR2021-00570
Page 1 of 74

1 MICHAEL A. JACOBS (CA SBN 111664)
mjacobs@mofo.com

2 RICHARD S.J. HUNG (CA SBN 197425)
rhung@mofo.com

3 FRANCIS C. HO (CA SBN 247426)
fho@mofo.com

4 ERIC W. OW (CA SBN 252921)
eow@mofo.com

5 MORRISON & FOERSTER, LLP
6 425 Market Street

7 San Francisco, CA 94105-2482

8 Telephone: (415) 268-7000

Facsimile: (415) 268-7522

9 BITA RAHEBI (CA SBN 209351)

brahebi@mofo.com

10 MORRISON & FOERSTER LLP

11 555 West Fifth Street, Suite 3500

Los Angeles, CA 90013-1024

12 Telephone: (213) 892-5200

13 Facsimile: (213) 892-5454

14 *Attorneys for Defendant Apple Inc.*

15 UNITED STATES DISTRICT COURT

16 CENTRAL DISTRICT OF CALIFORNIA

17 SOUTHERN DIVISION

18 ANCORA TECHNOLOGIES, INC.,

19 Plaintiff,

20 v.

21 APPLE INC.,

22 Defendant.

23 APPLE INC.,

24 Counterclaimant,

25 v.

26 ANCORA TECHNOLOGIES, INC.,

27 Counterdefendant.

Case No. 2:10-cv-10045-AG-MLG

**APPLE INC.'S N.D. CAL. PATENT
L.R. 3-3 DISCLOSURES**

28 sf-3054790

Ancora's Exhibit 2004
HTC v. Ancora IPR2021-00570

Page 2 of 74

1 Pursuant to the Court's August 29, 2011 Order Re: Stipulated Trial Schedule
2 (D.I. 40), Apple Inc. hereby provides its N.D. Cal. Patent L.R. 3-3 Disclosures
3 ("Invalidity Contentions") for U.S. Patent No. 6,411,941.

4 By providing these Invalidity Contentions, Apple does not waive any
5 applicable privilege or immunity, including the attorney-client privilege or work
6 product doctrine. Apple predicates the Invalidity Contentions, in part, on the claim
7 constructions suggested by Ancora's September 14, 2011 Discovery Order
8 Disclosures and Disclosures Pursuant to Patent Rules 3-1 and 3-2 ("Infringement
9 Contentions"). Accordingly, these Invalidity Contentions should not be read as
10 representing or otherwise reflecting Apple's final positions regarding the proper
11 interpretation of the claims. Ancora has asserted in its Infringement Contentions
12 that Apple's iPhone, iPod Touch, iPad and Apple TV infringe Claims 1-3 and 5-17
13 of the '941 patent ("Asserted Claims"). These Invalidity Contentions address only
14 the Asserted Claims.

15 Apple bases these Invalidity Contentions on information reasonably available
16 to it at this time. The significant deficiencies in Ancora's Infringement Contentions
17 and other discovery responses have made it difficult for Apple to understand
18 Ancora's infringement and claim construction positions, and those positions
19 necessarily inform Apple's invalidity positions.¹ Apple's investigation of Ancora's
20 claims and the prior art is ongoing. Apple incorporates by reference the
21 Preliminary Invalidity Contentions of Microsoft and PC Company Defendants in
22 *Ancora Technologies, Inc. v. Toshiba Am. Info. Sys., Inc.*, No. 2:09-cv-00270-MJP
23 (W.D. Wash.), attached as Exhibit A. Apple reserves the right to supplement or
24 amend these Invalidity Contentions in the future, particularly in response to any
25 supplementation by Ancora of its infringement contentions to clarify its theories.

26
27 ¹ See October 26, 2011 letter to Ancora's counsel outlining deficiencies.
28

A. Patent L.R. 3-3(a)

Apple identifies prior art publications and patents that anticipate or render obvious one or more of the limitations of the Asserted Claims in Table A below.

Author	Non-Patent Publication	Publication Date
White et al.	ABYSS: A Trusted Architecture for Software Protection, IEEE Transactions on Software Engineering, Vol. 16, No. 6, pp. 38-51 ("White 1990") (Ex. 1)	June 1990
Tygar et al.	Dyad: A System for Using Physically Secure Coprocessors, CMU-CS-94-140R, Carnegie Mellon University ("Tygar 1991") (Ex. 2)	May 4, 1991
Yee	Using Secure Coprocessors, Carnegie Mellon University, CMU-CS-94-149 ("Yee 1994") (Ex. 3)	May 1994
Clark et al.	BITS: A Smartcard Protected Operating System, Communications of the ACM, Vol. 37, No. 11, pp. 68-70; 94 ("Clark 1994") (Ex. 4)	Nov. 1994
Yee et al.	Secure Coprocessors in Electronic Commerce Applications, Proceedings of the 1st USENIX Workshop on Electronic Commerce, pp. 155-170 ("Yee 1995") (Ex. 5)	July 1995
Arbaugh et al.	A Secure and Reliable Bootstrap Architecture, Dept. of Comp. & Info. Sci. Tech. Reports, U. Penn. ("Arbaugh 1996") (Ex. 6)	1996
AMI et al.	Desktop Management BIOS Specification, Version 2.0 ("DMI BIOS Specification") (Ex. 7)	March 6, 1996
Arbaugh et al.	A Secure and Reliable Bootstrap Architecture, SP '97 Proceedings of the 1997 IEEE Symposium on Security and Privacy, pp. 66-71 ("Arbaugh 1997") (Ex. 8)	1997
Inventor	Patent Number	Issue Date
Hellman	U.S. 4,658,093 ("Hellman Patent") (Ex. 9)	Apr. 14, 1987
Joshi	U.S. 4,688,169 ("Joshi Patent") (Ex. 10)	Aug. 18, 1987
Allen et al.	U.S. 4,757,533 ("Allen Patent") (Ex. 11)	July 12, 1988
Karp	U.S. 4,866,769 ("Karp Patent") (Ex. 12)	Sep. 12, 1989
Waite	U.S. 5,103,476 ("Waite 476 Patent") (Ex. 13)	Apr. 7, 1992

1	Waite	U.S. 5,222,134 ("Waite 134 Patent") (Ex. 14)	Jun. 22, 1993
2	Smyth	U.S. 5,325,430 ("Smyth Patent") (Ex. 15)	June 28, 1994
3	Ewertz	U.S. 5,371,876 ("Ewertz Patent") (Ex. 16)	Dec. 6, 1994
4	Davis	U.S. 5,473,692 ("Davis 692 Patent") (Ex. 17)	Dec. 5, 1995
5	Richardson	U.S. 5,490,216 ("Richardson Patent") (Ex. 18)	Feb. 6, 1996
6	Schull	U.S. 5,509,070 ("Schull Patent") (Ex. 19)	Apr. 16, 1996
7	Morisawa et al.	U.S. 5,537,544 ("Morisawa Patent") (Ex. 20)	July 16, 1996
8	Davis et al.	U.S. 5,568,552 ("Davis 552 Patent") (Ex. 21)	Oct. 22, 1996
9	Christenson et al.	U.S. 5,579,522 ("Christenson Patent") (Ex. 22)	Nov. 26, 1996
10	McCarty	U.S. 5,666,411 ("McCarty Patent") (Ex. 23)	Sep. 9, 1997
11	Lewis	U.S. 5,734,819 ("Lewis Patent") (Ex. 24)	Mar. 31, 1998
12	O'Connor et al.	U.S. 5,745,568 ("O'Connor Patent") (Ex. 25)	Apr. 28, 1998
13	Davis	U.S. 5,844,986 ("Davis 986 Patent") (Ex. 26)	Dec. 1, 1998
14	Clark	U.S. 5,892,902 ("Clark Patent") (Ex. 27)	Apr. 6, 1999
15	Chou et al.	U.S. 5,892,906 ("Chou Patent") (Ex. 28)	Apr. 6, 1999
16	Labatte et al.	U.S. 5,901,311 ("Labatte 311 Patent") (Ex. 29)	May 4, 1999
17	Labatte et al.	U.S. 5,913,057 ("Labatte 057 Patent") (Ex. 30)	June 15, 1999
18	Griswold	U.S. 5,940,504 ("Grisworld Patent") (Ex. 31)	
19	Beelitz	U.S. 5,944,820 ("Beelitz Patent") (Ex. 32)	Aug. 31, 1999
20	Okada	U.S. 6,049,670 ("Okada Patent") (Ex. 33)	
21	Osborn	U.S. 6,026,293 ("Osborn Patent") (Ex. 34)	Feb. 15, 2000
22	Miller	U.S. 6,038,320 ("Miller Patent") (Ex. 35)	Mar. 14, 2000
23	Mirov et al.	U.S. 6,138,236 ("Mirov Patent") (Ex. 36)	Oct. 24, 2000
24	Fieres et al.	U.S. 6,148,083 ("Fieres Patent") (Ex. 37)	Nov. 14, 2000
25	Schwartz et al.	U.S. 6,153,835 ("Schwartz Patent") (Ex. 38)	Nov. 28, 2000
26	Arbaugh et al.	U.S. 6,185,678 ("Arbaugh Patent") (Ex. 39)	Feb. 6, 2001
27	Misra et al.	U.S. 6,189,146 ("Misra Patent") (Ex. 40)	Feb. 13, 2001
28	Saunders	U.S. 6,209,099 ("Saunders Patent") (Ex. 41)	Mar. 27, 2001
	Pearce et al.	U.S. 6,243,468 ("Pearce Patent") (Ex. 42)	Jun. 5, 2001
	Cotichini et al.	U.S. 6,269,392 ("Cotichini Patent") (Ex. 43)	July 31, 2001

Pavlin et al.	U.S. 6,523,119 (“Pavlin Patent”) (Ex. 44)	Feb. 18, 2003
Hannah	U.S. 6,735,696 (“Hannah Patent”) (Ex. 45)	May 11, 2004

Table A: Pat. L.R. 3-3(a) Identification of References

Furthermore, Apple identifies the following products that may constitute prior use, offer for sale, sale, or knowledge under 35 U.S.C. § 102.

- Systems and methods invented, designed, developed and/or in public use or on sale related to BIOS locking: Many PC manufacturers utilized a BIOS locking concept by which media used to restore PC systems would install only if certain information was present in the target non-volatile memory holding the PC’s BIOS. Specifically, PC manufacturers would place information in such a non-volatile memory containing the PC’s BIOS. When the system restore CD was loaded, the installation program would not install the restore programs (e.g. operating system) if the installation program could not confirm the presence of the proper information in the non-volatile memory/BIOS. PC manufacturers would implement this BIOS Locking concept using their own software solutions. PC manufacturers also implemented BIOS Locking in connection with recovery partitions on a hard disk.
- Systems and methods invented, designed, developed and/or in public use or on sale related to ABYSS. A publication describing ABYSS is identified in Table A (White et al. article).
- Systems and methods invented, designed, developed and/or in public use or on sale related to Dyad and/or its extensions: Publications describing Dyad are identified in Table A (Tygar 1991, Yee 1994, Yee 1995).
- Systems and methods invented, designed, developed and/or in public use or on sale related to DMI BIOS: Publications and patents

describing the DMI BIOS are identified in Table A (DMI BIOS Specification & Labatt et al. patents).

- Systems and methods invented, designed, developed and/or in public use or on sale related to AEGIS: Publications and patents describing AEGIS are identified in Table A and Exhibit 46 (Arbaugh et al. articles and patent).
- Systems and methods invented, designed, developed and/or in public use or on sale related to the Citadel coprocessor: Publications and patents describing Citadel are identified in Table A (Yee articles).

Although the articles and patents corresponding to the above products disclose how the products anticipate or render obvious the Asserted Claims, the products themselves may contain other instances of anticipating or rendering obvious the Asserted Claims. Where Apple cites to articles and patents referring to one version of the product, including those products not specifically identified above, Apple incorporates and relies on all versions of the same product. As discovery is ongoing, Apple reserves the right to identify other locations within the product where the Asserted Claims are disclosed, or other documents and code evidencing the structure and operation of these products.

Depending on the Court's conclusions as to the constructions of the Asserted Claims, Apple may identify or rely upon additional references, either individually or in combination, as anticipating or rendering obvious the Asserted Claims. Some of these potential additional references appear in Exhibit 46. Apple reserves the right to amend its invalidity contentions or these references depending on the claim construction and infringement positions Ancora may take.

B. Patent L.R. 3-3(b)

The Asserted Claims of the '941 patent are invalid in light of the prior art identified above. To the extent that any of the prior art references identified above do not anticipate the Asserted Claims of the '941 patent, their combination with the

1 knowledge of one of ordinary skill in the art and/or other prior art disclosing the
2 allegedly missing limitations renders the asserted claims of the '941 patent obvious.

3 Because the '941 patent simply arranges old elements with each performing
4 the same function it had been known to perform and yields no more than what one
5 would expect from such an arrangement, the combination is obvious. *See KSR Int'l*
6 *Co. v. Teleflex Inc.*, 550 U.S. 398, 419-421 (2007). Moreover, since there were a
7 finite number of predictable solutions, a person of ordinary skill in the art had good
8 reason to pursue the known options. *Id.* In particular, the Supreme Court stated
9 that courts should “look to interrelated teachings of multiple patents; the effects of
10 demands known to the design community or present in the marketplace; and the
11 background knowledge possessed by a person having ordinary skill in the art, all in
12 order to determine whether there was an apparent reason to combine the known
13 elements in the fashion claimed by the patent at issue.” *Id.* at 418.

14 In the prior art, there were well-recognized design needs and market
15 pressures to restrict the use of unauthorized software. For instance, the Joshi Patent
16 recognized this need as it sought to provide “an improved computer software
17 security system which prevents a program from executing on computers other than
18 an authorized machine” (e.g., col. 2:25-29). The prior art suggests the same or
19 similar solutions to these issues. Using the BIOS to provide security to software
20 was common in the marketplace. For instance, the Miller Patent taught using the
21 “BIOS flash memory” (e.g., col. 2:59-3:7) and the Chou Patent taught using the
22 “BIOS EEPROM” (e.g., col. 8:18-23; Fig. 3.). Moreover, the O'Connor Patent
23 taught that an “identifier is permanently programmed into a segment or multiple
24 segments of a BIOS memory” (e.g., col. 3:34-49), and the Hannah Patent taught
25 that “the BIOS certifies the operating system” of a computer (e.g., col. 2:54-3:41).
26 Additionally, Arbaugh 1996 recognized that the BIOS could “contain the
27 verification code, and public key certificates” (e.g., p. 66). Indeed, the Cotichini
28 Patent recognized that a “security system may be embodied in internal memory

1 devices such as ROM BIOS” (e.g., Abstract), and the Misra Patent taught that
2 software licenses could be organized in a license cache that is “kept in persistent
3 (non-volatile) storage” (e.g., 12:8-16). Thus, it would have been obvious to one of
4 ordinary skill in the art to combine the teachings in any of these references.

5 Such similar solutions and interrelated teachings provided ample reason to
6 combine the prior art elements as taught in any of these references with each other
7 or the features in other prior art, such as the Davis Patents, which taught a security
8 solution involving an “integrated circuit component including a memory element
9 for internally storing a unique digital certificate” (e.g., Davis 692 Patent, col. 2:58-
10 65). Likewise, any of these references could be combined with the Lewis Patent,
11 which taught “using non-volatile memory where critical data is written” to monitor
12 security of software (e.g., col. 2:66-3:3), the Clark Patent, which taught that
13 “critical information such as boot sector information [], file integrity information
14 []and authentication information []may be stored in the EEPROM” (e.g., col. 3:24-
15 44), or the Osborn Patent, which taught authentication of values stored in EEPROM
16 (e.g., col. 8:21-42). Because of the inter-related teachings of these prior art
17 references, it would have been obvious to one of ordinary skill in the art to combine
18 any of these references.

19 Indeed, the ’941 patent acknowledges that it was well known in the art to
20 “validate authorized software usage by writing a license signature onto the
21 computer’s volatile memory (e.g., hard disk)” (1:19-21). Using the BIOS and/or
22 non-volatile memory to “validate authorized software usage” would have been
23 obvious to one of ordinary skill in the art. For instance, in addition to the security
24 techniques of the prior art above, the security methods taught in the Lewis Patent
25 used “electronic security information stored in the non-volatile memory” (e.g., col.
26 3:6-15). Thus, it would have been obvious to one of ordinary skill in the art to
27 combine any of the above references.
28

1 The prior art references themselves also suggest their use in a variety of
2 applications and combinations, further supporting a finding of obviousness. For
3 example, the Yee references teach using “a secure processor” that can provide
4 “software copy protection” as well as “greater protection for customers in point-of-
5 sale applications” (e.g., Yee 1995, p. 155).” The Allen Patent taught a “tamper-
6 proof manner of protecting user-access and file-access” (e.g., Abstract). Tygar
7 1991 implemented a solution for “protecting the integrity of publicly accessible
8 workstations” and “copy protection” (e.g., Abstract). As another example, the
9 Arbaugh references recognized the need for “a secure bootstrap process” by
10 “ensuring the integrity of bootstrap code” (e.g., Arbaugh 1996 at p. 1). Similarly,
11 the Beelitz Patent taught a “method for providing a modifiable partition boot
12 record” (e.g., Abstract). Thus, it would have been obvious to one of ordinary skill
13 in the art to combine the teachings of any of these prior art references with each
14 other or the features in other prior art, such as the Richardson Patent, which
15 described a solution to protect commercially available software (e.g., col. 1:10-
16 2:24) or the Hellman Patent, which described a solution for a “secure software
17 distribution system” (e.g., col. 1:4-7). Indeed, the background of the ’941 patent
18 recites that “[n]umerous methods have been devised for identifying and restricting
19 of an unauthorized software program’s operation” (col. 1:11-13). Moreover, the
20 interrelated teachings of the prior art as described in Exhibits 1-45 support a finding
21 of obviousness.

22 The nature of the problem that was sought to be solved also would suggest
23 that the ’941 patent was obvious to one of the ordinary skill in the art. For instance,
24 the art recognized using secured components that are not “vulnerable to attack at
25 the hands of skilled system’s programmers (e.g., ‘hackers’)” as described in the
26 ’941 patent (col. 1:21-24). In addition to the many references described above
27 which taught using the BIOS, non-volatile memory, secure processors, or a secure
28 bootstrap process for enhanced software security, Clark 1994 taught the use of

1 “secure processors” and “smartcards” that are “resistant to attack, as they need not
2 depend upon potentially vulnerable external resources” (p. 66-67). Likewise,
3 Abyss taught a “protected processor” with “secure memory for storage of Rights-
4 To-Execute” that govern software protection (e.g., p. 39). The Pavlin Patent taught
5 an “EEPROM address space” that included “a license address space” (e.g., col.
6 5:33-56). Similarly, the Joshi Patent taught using codes stored in PROM to verify
7 authorized software (e.g., col. 3:59-63), and the Mirov Patent taught a flash PROM
8 that “is divided into two main sections: the authentication section . . . and a
9 programmable section” (e.g., col. 3:56-4:7). Other examples include Tygar 1991,
10 which taught that “cryptographic checksums of images must be stored in the secure
11 coprocessor’s NVM and protected against modification” (e.g., p. 9) and Yee 1995,
12 which taught “a secure coprocessor” that has “memory inaccessible to all entities
13 except the secure coprocessor itself – enough private non-volatile memory to store
14 the secrets” (e.g., p. 166). Thus, it would have been obvious to one of ordinary
15 skill in the art to combine the teachings in any of these references.

16 Additionally, it would have been obvious to one of ordinary skill in the art to
17 combine the teachings in any of the above references with the teachings in other
18 references related to modifying or expanding the BIOS. Such references include
19 the DMI BIOS Specification and the Labatte Patents, which taught methods of
20 utilizing the computer BIOS, the Smyth Patent, which taught “[m]odifications to
21 the standard BIOS” (e.g., col. 15:1-28), the Ewertz Patent, which taught “expanding
22 the memory capacity for the BIOS” (e.g., col. 3:21-46), or the Christenson Patent,
23 which taught a flash memory that contained an updateable BIOS area (e.g., col.
24 3:30-36). Similarly, any of these references could be combined with the Chou
25 Patent, which taught a “security function” that was “stored in the BIOS memory”
26 (e.g., col. 2:14-23) or the Miller Patent, which taught matching key codes “assigned
27 to the computer [] and stored in BIOS flash memory” (e.g., col. 4:7-29).

28

1 To add another layer of protection to software, one of ordinary skill in the art
2 would have added encryption and cryptography to the teachings of any of the above
3 references based on his background knowledge or based on the methods taught in,
4 for example, the Fieres Patent (e.g., col. 7:49-56), the Saunders Patent (e.g., col.
5 2:52-60), or the Mirov Patent (e.g., col. 4:27-55). Additionally, any of the above
6 references could be combined with the Osborn Patent which taught a method that
7 “authenticates the signed value pair . . . stored in the EEPROM [which] may
8 involve authenticating the signed valid hash value by processing it with a public
9 key []and then comparing the result with the unsigned hash value” (e.g., col. 8:21-
10 42) or the Davis 692 Patent, which taught “a non-volatile memory element [], such
11 as flash memory, which stores appropriate cryptographic algorithms such as ‘RSA’
12 and ‘DES’, the public/private key pair [], [and] a digital certificate for verifying the
13 authenticity of the key pair (labeled ‘DC’))” (e.g., col. 6:62-7:8).

14 Furthermore, one of ordinary skill in the art would have implemented the
15 protection methods of the above identified references with a remote server, based
16 on his background knowledge. Additionally, one of ordinary skill in the art would
17 have implemented the protection methods of the above identified references by
18 using a remote “licensing processor” such as that taught in the Schull Patent (e.g.,
19 col. 5:1-18) or a remote “registration computer” such as that taught in the Waite
20 Patents (e.g., Waite 476 Patent, at Abstract). Likewise, the Richardson Patent
21 taught communicating with a remote “registration authority” to check authorized
22 use of software (e.g., col. 7:36-51), and the Clark Patent taught security methods
23 via a “remote host computer” (e.g., col. 6:32-36). Other relating teachings include,
24 for instance, the Griswold Patent, which disclosed remote authorization of a
25 licensed product via a “communications network” (e.g., col. 5:44-52) and the
26 Okada Patent, which discloses that a user connecting to a remote “host computer”
27 to register software (e.g., col. 8:42-46). The methods taught in the above prior art
28 could be combined with, for instance, the protection methods of Tygar 1991, which

1 teach a “communication server” (e.g., p. 25) or the protection methods of Clark
2 Patent, which teach a “remote host computer” (e.g., col. 6:1-23).

3 Also, one of ordinary skill in the art would have combined the methods in the
4 above prior art with his knowledge about generating license records from the
5 software information. Moreover, such a teaching is prevalent in the prior art and
6 one of ordinary skill in the art would have combined such prior art teachings with
7 other references. For instance, The Design and Implementation of Tripwire (Kim),
8 teaches generating records that include file attributes from the file metadata and file
9 itself. Indeed, Yee 1994 specifically references TripWire (e.g., p. 17). Likewise,
10 the above prior art references could be combined with a method to “hash the
11 concatenation of [a] product ID and hardware ID to produce [a] registration ID” to
12 send to a “registration authority” as taught by the Pearce Patent (e.g., col. 2:52-
13 3:15) or a method to use a “tariff file” that includes an “application ID,” “publisher
14 identification code” and an “encryption key,” as taught by the Archibald patent
15 (e.g., U.S. 5,825,883, at col. 6:12-27). The prior art could also be combined with
16 the “clearinghouse” and licenses used to protect software as taught by the Misra
17 Patent.

18 Moreover, one of ordinary skill in the art would have implemented the
19 protection methods of any of the above identified references using secure keys,
20 identifications, or passwords such as a “randomly determined CPU identification”
21 or a “unique source identification” as taught by the Karp Patent (e.g., col. 2:51-66),
22 a “system identifier . . . and a system sipher key” as taught by the McCarty Patent
23 (e.g., col. 7:31-8:3) or “authorization numbers” as taught by the Schwartz Patent
24 (e.g., col. 10:21-54). Similarly, any of the above references could also be combined
25 with the Morisawa Patent, which taught “a non-volatile password memory for
26 storing one or more registered passwords” (e.g., col. 3:4-23), the Schull Patent,
27 which taught a “nearly-unique ‘tattoo’” stored in non-volatile memory (e.g., col.
28

8:31-50) or the Labatte Patents, which taught that an “access key is passed to BIOS” (e.g., Labbatte 311 Patent, at col. 5:44-54).

Additionally, the above identified prior art references use those familiar elements for their primary or well known purposes in a manner well within the ordinary level of skill in the art. Accordingly, common sense and knowledge of the prior art render the claims invalid under either § 102 or § 103.

As explained above, a person of ordinary skill would have been motivated to combine the above prior art based on the nature of the problem to be solved, the teachings of the prior art, and the knowledge of persons of ordinary skill in the art. To the extent that Ancora challenges a combination of prior art with respect to a particular element, Apple reserves the right to supplement these contentions to further specify the motivation to combine the prior art. Apple may rely on cited or uncited portions of the prior art, other documents, and expert testimony to establish that a person of ordinary skill in the art would have been motivated to modify or combine the prior art so as to render the claims invalid as obvious. In addition to the combinations identified above, a non-exhaustive list of exemplary combinations of prior art that render obvious the Asserted Claims include:

- Miller Patent in view of one or more of the following: Yee 1995, Tygar 1991, Arbaugh 1996, Hannah Patent, Lewis Patent, Joshi Patent, Clark Patent, Chou Patent, Davis 692 Patent, Misra Patent, Schull Patent, Richardson Patent, Waite Patents, and/or Fierres Patent
- Yee 1995 in view of one or more of the following: Miller Patent, Tygar 1991, Arbaugh 1996, Hannah Patent, Clark Patent, Joshi Patent, Lewis Patent, Chou Patent, Davis 692 Patent, Misra Patent, Schull Patent, Richardson Patent, Waite Patents, and/or Fierres Patent
- Arbaugh 1996 in view of one or more of the following: Miller Patent, Tygar 1991, Yee 1995, Hannah Patent, Clark Patent, Joshi Patent, Lewis

Patent, Chou Patent, Davis 692 Patent, Misra Patent, Schull Patent,
Richardson Patent, Waite Patents, and/or Fieres Patent

- Clark Patent in view of one or more of the following: Miller Patent, Yee 1995, Tygar 1991, Arbaugh 1996, Hannah Patent, Joshi Patent, Lewis Patent, Chou Patent, Davis 692 Patent, Misra Patent, Schull Patent, Richardson Patent, Waite Patents, and/or Fieres Patent

- Lewis Patent in view of one or more of the following: Miller Patent, Yee 1995, Tygar 1991, Arbaugh 1996, Hannah Patent, Joshi Patent, Chou Patent, Clark Patent, Davis 692 Patent, Misra Patent, Schull Patent, Richardson Patent, Waite Patents, and/or Fieres Patent

- Tygar 1991 in view of one or more of the following: Lewis Patent, Miller Patent, Yee 1995, Arbaugh 1996, Hannah Patent, Joshi Patent, Chou Patent, Clark Patent, Davis 692 Patent, Misra Patent, Schull Patent, Richardson Patent, Waite Patents, and/or Fieres Patent

- Chou Patent in view of one or more of the following: Lewis Patent, Miller Patent, Yee 1995, Arbaugh 1996, Hannah Patent, Joshi Patent, Tygar 1991, Clark Patent, Davis 692 Patent, Misra Patent, Schull Patent, Richardson Patent, Waite Patents, and/or Fieres Patent

C. Patent L.R. 3-3(c)

Charts identifying where in each item of prior art each limitation of each Asserted Claim is found, is attached hereto as Exhibits 1-43. Apple cites the most relevant portions of the identified prior art references. However, other portions of the identified prior art may additionally disclose one or more limitations of the Asserted Claims.

D. Patent L.R. 3-3(d)

Depending on the Court's construction of the Asserted Claims, the Asserted Claims are invalid for lack of written description and/or enablement under 35 U.S.C. § 112 ¶ 1 and for failing to particularly point out and distinctly claim the

1 subject matter that the applicant regards as his invention under 35 U.S.C. § 112 ¶ 2.
 2 In light of the deficiencies in Ancora's Infringement Contentions, Apple reserves
 3 the right to supplement these Invalidity Contentions to further identify bases for
 4 invalidity under 35 U.S.C. § 112, ¶¶ 1 & 2. Apple identifies below the grounds
 5 upon which the Asserted Claims are invalid based on indefiniteness under 35
 6 U.S.C. § 112 ¶ 2 and lack of written description and enablement under 35 U.S.C.
 7 § 112 ¶ 1.

8 **1. Indefiniteness under 35 U.S.C. § 112 ¶ 2**

- 9 • **“volatile memory” and “non-volatile memory”** – Claim 1 recites both
 10 “volatile memory” and “non-volatile memory.” “Volatile memory”
 11 generally refers to memory that loses the data it holds when the power
 12 to the memory is removed. The '941 patent, however, states that
 13 “volatile memory” includes at least a “hard disk,” (col. 1:19-21), which
 14 does not lose the data it holds when the power to the hard disk is
 15 removed. Given the conflicting use of the terms “volatile memory”
 16 and “non-volatile memory,” it is uncertain what falls within the scope
 17 of “volatile memory” and further uncertain what falls within the scope
 18 of “non-volatile memory.” Thus, claim 1 and all Asserted Claims that
 19 depend from claim 1 are indefinite.
- 20 • **“selecting a program residing in the volatile memory”** – Claim 1
 21 recites “selecting a program residing in the volatile memory.” The
 22 '941 patent states “selecting a program” includes the step of
 23 “establishing a licensed-software-program in the volatile memory”
 24 (col. 6:7-10). It is unclear how a skilled artisan would “establish[] a
 25 licensed-software-program in the volatile memory” if such a program
 26 is already “residing in the volatile memory.” Given the apparent
 27 conflicting definition of this term, claim 1 and all Asserted Claims that
 28 depend from claim 1 are indefinite.

- 1 • **“agent”** – Claim 1 recites an “agent.” The term “agent” does not
2 appear anywhere in the written description of ’941 patent.
3 Accordingly, claim 1 and all Asserted Claims that depend from claim 1
4 are indefinite.
- 5 • **“the unique key”** – Claim 5 recites “the unique key.” There is no
6 antecedent basis for “the unique key.” Accordingly, claim 5 is
7 indefinite.
- 8 • **“establishing a licensed-software-program in the volatile memory
9 area of the computer”** – Claim 6 recites “establishing a licensed
10 software-program in the volatile memory area of the computer.” It is
11 vague as to what constitutes “establishing a licensed-software-program
12 in the volatile memory area of the computer.” This vagueness is
13 compounded by the indefiniteness of “volatile memory area,” as
14 discussed above. Accordingly, claim 6 is indefinite.
- 15 • **“wherein the step of using the agent to set up the verification
16 record”** – Claim 14 recites “wherein the step of using the agent to set
17 up the verification record.” There is no antecedent basis for “the
18 verification record.” Accordingly, it is uncertain what this step refers
19 to, rendering claim 14 and all Asserted Claims that depend from claim
20 14 indefinite.
- 21 • **“first non-volatile memory area of the computer”** – Claims 7 and 13
22 (and their dependents) recite a “first non-volatile memory area of the
23 computer.” It is vague as to whether this refers to the non-volatile
24 memory area of claim 1 or a different one. Accordingly, these claims
25 are indefinite.
- 26 • **“the erasable second non-volatile memory area of the BIOS”** –
27 Claim 16 depends on claim 13 and further recites “the erasable second
28 non-volatile memory area of the BIOS.” This increases the vagueness

1 already present in claim 13 because there is no antecedent basis for
 2 “the erasable second non-volatile memory area of the BIOS.” Further,
 3 it is unclear whether this memory refers to the “erasable, non-volatile
 4 memory area of the BIOS” in claim 1, which should be “first,” or the
 5 “first non-volatile memory area” of claim 13, or a different “erasable
 6 second non-volatile memory area of the BIOS.” Thus, claim 16 is
 7 indefinite.

8 **2. Lack of Written Description under 35 U.S.C. § 112 ¶ 1**

- 9 • **“license authentication bureau”** – The specification states that the
 10 “license authentication bureau . . . may form part of the computer.”
 11 (col. 6:1-3). To the extent that any claim includes within its scope such
 12 a license authentication bureau, including but not limited to claims 1-2
 13 and 5-17, the claim is invalid for lack of written description because
 14 the patent fails to adequately describe a bureau that forms part of the
 15 computer. The patent further does not adequately describe the
 16 “establishing” of a license authentication bureau, as recited by claims 2
 17 and 3.
- 18 • **“agent”** – Claim 1 recites an “agent.” The term “agent” does not
 19 appear anywhere in the written description of ’941 patent.
 20 Accordingly, claim 1 and all Asserted Claims that depend from claim
 21 11 do not comply with the written description requirement.
- 22 • **“set up a verification structure”** – Claim 1 recites “set up a
 23 verification structure.” The ’941 patent discloses that this includes the
 24 steps of “establishing or certifying the existence of a pseudo-unique
 25 key in the first non-volatile memory area; and establishing at least one
 26 license-record location in the first or the second nonvolatile memory
 27 area.” (Col. 6:18-22). To the extent that Ancora contends that
 28 “set[ting] up a verification structure” does not include the above steps,

claim 1 and all Asserted Claims that depend from claim 1 do not comply with the written description requirement.

- **“volatile memory”** and **“non-volatile memory”** – Claim 1 recites both “volatile memory” and “non-volatile memory.” As discussed above, these terms are indefinite because of the uncertain use of “volatile memory” and “non-volatile memory.” To the extent Ancora contends the definition of “volatile memory” is definite, the specification does not provide an adequate description to show that the inventor possessed “volatile memory” that would include a hard disk. Similarly, to the extent Ancora contends the definition of “non-volatile memory” is definite, the specification does not provide an adequate description to show the inventors possessed “non-volatile memory.” Thus, claim 1 and all Asserted Claims that depend from claim 1 do not comply with the written description requirement.

3. **Lack of Enablement under 35 U.S.C. § 112 ¶ 1**

- **“license authentication bureau”** – The specification states that the “license authentication bureau . . . may form part of the computer.” (col. 6:1-3). To the extent that any claim includes within its scope such a license authentication bureau, including but not limited to claims 1-2 and 5-17, the claim is invalid for lack of enablement because the patent lacks adequate disclosure to enable the use of the full scope of the claims. The patent further lacks adequate disclosure to enable the “establishing” of a license authentication bureau, as recited by claims 2 and 3.
- **“agent”** – Claim 1 recites an “agent.” The term “agent” does not appear anywhere in the written description of ’941 patent. The ’941 patent lacks disclosure to enable use of an “agent” in connection with the steps of the Asserted Claims. Accordingly, claim 1 and all

1 Asserted Claims that depend from claim 11 do not comply with the
2 enablement requirement.

- 3 • **“set up a verification structure”** – Claim 1 recites “set up a
4 verification structure.” The ’941 patent states this includes the steps of
5 “establishing or certifying the existence of a pseudo-unique key in the
6 first non-volatile memory area; and establishing at least one license-
7 record location in the first or the second nonvolatile memory area.”
8 (col. 6:18-22). To the extent that Ancora contends that “set[ting] up a
9 verification structure” does not include the above step, claim 1 and all
10 Asserted Claims that depend from claim 1 do not comply with the
11 enablement requirement.

12 **E. Patent L.R. 3-4(a-b)**

13 Apple produces or makes available for inspection and copying documents
14 required by Patent L.R. 3-4(a) & (b) that are in Apple’s possession, custody or
15 control. Documents at APL00007846-8608 include documents corresponding to
16 Patent L.R. 3-4(a). Moreover, Apple has made available for inspection source code
17 pursuant to Patent L.R. 3-4(a). Documents at APL00002937-7845 & 8609-10101
18 include documents corresponding to Patent L.R. 3-4(b).

1 Dated: October 28, 2011

Respectfully submitted,

2 **MORRISON & FOERSTER, LLP**

3 By: /s/ Francis Ho

4 Michael A. Jacobs
5 mjacobs@mofo.com
6 Richard S.J. Hung
7 rhung@mofo.com
8 Francis C. Ho
9 fho@mofo.com
10 425 Market Street
11 San Francisco, CA 94105-2482
12 T. (415) 268-7000
13 F. (415) 268-7522

14 Bitu Rahebi
15 brahebi@mofo.com
16 MORRISON & FOERSTER LLP
17 555 West Fifth Street, Suite 3500
18 Los Angeles, CA 90013-1024
19 T. (213) 892-5200
20 F. (213) 892-5454

21 Attorneys for Apple Inc.

CERTIFICATE OF SERVICE

I declare that I am employed with the law firm of Morrison & Foerster LLP, whose address is 425 Market Street, San Francisco, California 94105-2482; I am not a party to the this action, and I am over the age of eighteen years.

I further declare that on the date hereof a copy of:

APPLE INC.'S N.D. CAL. PATENT L.R. 3-3 DISCLOSURES

was served by Electronic Mail on the individuals below:

William Thompson	Mark A. Cantor
wthompson@brookskushman.com	mcantor@brookskushman.com
BROOKS KUSHMAN P.C.	John S. Le Roy
6701 Center Drive, Suite 610	jleroy@brookskushman.com
Los Angeles, CA 90045	Marc Lorelli
Tel.: (310) 348-8200	mlorelli@brookskushman.com
Fax: (310) 846-4799	John Rondini
	jrondini@brookskushman.com
	Leanne Scott
	lscott@brookskushman.com
	1000 Town Center, Twenty-Second
	Floor
	Southfield, MI 48075
	Tel.: (248) 358-4400
	Fax: (248) 358-3351

I declare under penalty of perjury that the above is true and correct.

Executed at San Francisco, California, this 28th day of October, 2011.

Francis C. Ho
(typed)

/s/ Francis C. Ho
(signature)

Exhibit 1**Claim Chart for U.S. Patent No. 6,411,941 based on White et al (ABYSS: A Trusted Architecture for Software Protection)**

Element	Claim Language	White et al.
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	Abstract (“ABYSS (A Basic Yorktown Security System) is an architecture for the trusted execution of application software. It supports a uniform security service across the range of computing systems. The use of ABYSS discussed in this paper is oriented towards solving the software protection problem, especially in the lower end of the market. Both current and planned software distribution channels are supportable by the architecture, and the system is nearly transparent to legitimate users. A novel use-once authorization mechanism, called a token, is introduced as a solution to the problem of providing authorizations without direct communication. Software vendors may use the system to obtain technical enforcement of virtually any terms and conditions of the sale of their software, including such things as rental software. Software may be transferred between systems, and backed up to guard against loss in case of failure. We discuss the problem of protecting software on these systems, and offer guidelines to its solution. ABYSS is shown to be a general security base, in which many security applications may execute.”) Pg. 39 (“A protected processor constitutes a minimal, but complete, computing system. It contains a processor, a real-time clock, a random or pseudo-random number generator, and sufficient memory to store protected parts of applications while they execute. It also contains secure memory for storage of Rights-To-Execute. This storage retains its contents even when the system power is off. The protected processor is a logically, physically, and procedurally secure unit. It is logically secure, in that an application cannot directly access the supervisor process, or the protected part of any other application, to violate their protection. It is physically secure (which is indicated by the heavy box in Fig. 1), in that it is contained in a tamper-resistant package It is procedurally secure in that the

sf-3015903

1

Exhibit 2**Claim Chart for U.S. Patent No. 6,411,941 based on Tygar “Dyad: A System for Using Physically Secure Coprocessors” (1991)**

Element	Claim Language	Tygar
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	p. 1 (“The Dyad project at Carnegie Mellon University is using physically secure coprocessors to achieve new protocols and systems addressing a number of perplexing security problems. These coprocessors can be produced as boards or integrated circuit chips and can be directly inserted in standard workstations or PC-style computers.”) p. 4 (“What do we mean by the term <i>secure coprocessor</i>? A secure coprocessor is a hardware module containing (1) a CPU, (2) ROM, and (3) NVM (non-volatile memory). This hardware module is physically shielded from penetration, and the I/O interface to this module is the only means by which access to the internal state of the module can be achieved.”) p. 9 (“One model which solves the host integrity problem is that of using a secure coprocessor to perform the necessary integrity checks. Because of the privacy and integrity guarantees on secure coprocessor memory and processing, we can use a secure coprocessor to check the integrity of the host’s state at bootup and have confidence in the results. At boot time, the secure coprocessor is the first to gain control of the system and can decide whether to allow the host CPU to continue by first checking the disk-resident bootstrap program, operating system kernel, and all system utilities for evidence of tampering.”) p. 9 (“The cryptographic checksums of system images must be stored in the secure coprocessor’s NVM and protected against modification, and, depending on the cryptographic checksum algorithm chosen, exposure. Of course, tables of cryptographic checksums can be paged out to host memory or disk after first checksumming and encrypting them within the secure coprocessor; this can be handled as an extension to normal virtual memory paging.”)

Exhibit 3**Claim Chart for U.S. Patent No. 6,411,941 based on Yee (“Using Secure Coprocessors”)**

Element	Claim Language	Yee
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	p. 39 (“The PS/2 host contains an Intel i386 CPU running at 16 MHz, 16 megabytes of RAM, and a microchannel system bus. The Citadel coprocessor contains an Intel i386SX CPU, also running at 16 MHz; one megabyte of ‘scratch’ volatile RAM; 64 kilobytes of battery backed secure RAM; bootstrap EPROM with a simple monitor program; and 64 kilobytes of second-stage bootstrap EEPROM; an IBM produced DES chip with a theoretical throughput of 30 megabytes per second.”) (footnotes omitted) “The secure coprocessor ensures that the system securely boots; after booting, a secure coprocessor aids the host operating system by providing security functions. . . . When the host system is up and running, a secure coprocessor provides various security services to the host operating system: * integrity verification of any stored data (by secure checksums); * data encryption to boost storage media natural security (see section 2.4); and * encrypted communication channels (key exchange, authentication, private key encryption, etc.).” (Yee, p. 36) (footnotes omitted) p. 49 (“The Citadel coprocessor system provides 64 kilobytes of battery-backed memory (secure RAM / non-volatile memory) protected by intrusion detection circuitry.”)
1-A	selecting a program residing in the volatile memory,	See Fig. 3.1.

Exhibit 4**Claim Chart for U.S. Patent No. 6,411,941 based on Clark, BITS: A Smartcard Protected Operating System**

Element	Claim Language	Clark
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	p. 66 (“The capability of a smartcard is defined by its IC. As the name implies, an integrated circuit consists of multiple components combined within a single chip. Some possible components are a microprocessor, nonstatic random access memory (RAM), read only memory (ROM), erasable programmable read only memory (EPROM), other nonvolatile memory (memory that retains its state when current is removed) such as electrically erasable programmable read only memory (EEPROM), and special purpose coprocessor(s) (see Figure 1).”) p. 66 (“Secure processors as a technology offer a great deal for applications that require enhanced computer security. In general, these processors contain not only computational capability, but memory capacity as well [17]. This self-containment makes secure coprocessors resistant to attack, as they need not depend upon potentially vulnerable external resources.”) p. 67 (“A user-supplied smartcard stores the user’s private key and in addition provides service calls for digital signatures and encryption so that all operations involving the private key are performed on the card.”) p. 67-68 (“To prevent deliberate sabotage as well as inadvertent catastrophe, some means of access control is desirable in a shared computer or network environment.”) p. 68 (“The Boot Integrity Token System (BITS) is being developed to provide computer boot integrity and enforce access control [4].”) Figure 1.

sf-3033588

1

Exhibit 5

Claim Chart for U.S. Patent No. 6,411,941 based on Yee, “Secure Coprocessors in Electronic Commerce Applications”

Element	Claim Language	Yee
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	p. 155 (“This paper argues for a more powerful model – a secure <i>coprocessor</i> – that can run a small operating system, run application programs, and also keep secure storage for cryptographic keys and balance information. Secure coprocessors are tamper-proof sealed devices that have a processor, memory storage, and (optional) fast crypto-support.”) (emphasis in original) p. 155 (“We have built a system called Dyad, on top of a port of the Mach 3.0 microkernel to the IBM Citadel secure coprocessor. This paper describes the abstract architecture of Dyad and a general discussion of secure coprocessor implementations of a variety of electronic commerce applications: * Copy protection for software . . .”) (citations omitted) p. 166 (“Thus a secure coprocessor must have memory inaccessible to all entities except the secure coprocessor itself – enough private non-volatile memory to store the secrets, plus private (possibly volatile) memory for intermediate calculations in running protocols.”) p. 167 (“A small, simple security kernel is needed for the secure coprocessor . . . [T]he secure coprocessor kernel must provide separate address spaces for vendor and user code in the secure coprocessor – even if we implicitly trust vendor and user code, providing separate address spaces helps isolate the effects of programming errors. . . . <i>Key management</i> includes management of authentication secrets, cryptographic keys, and system fingerprints of executables and data. With the limited number of services needed, we can easily envision using a microkernel such as Mach 3.0, the NT executive, or QNX. We only need to add a communications server and include a key management service to manage secure non-volatile key memory.”) (emphasis in original) (citations omitted).

Exhibit 6**Claim Chart for U.S. Patent No. 6,411,941 based on Arbaugh, A Secure and Reliable Bootstrap Architecture**

Element	Claim Language	Arbaugh
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	p. 65 (“We have designed AEGIS, a secure bootstrap process. AEGIS increases the security of the boot process by ensuring the integrity of bootstrap code. It does this by constructing a chain of integrity checks, beginning at power-on and continuing until the final transfer of control from the bootstrap components to the operating system itself. The integrity checks compare a computed cryptographic hash value with a stored digital signature associated with each component.”) p. 67 (“We have divided the boot process into several levels to simplify and organize the AEGIS BIOS modifications, as shown in figure 3. Each increasing level adds functionality to the system, providing correspondingly higher levels of abstraction. The lowest level is Level 0. Level 0 contains the small section of trusted software, digital signatures, public key certificates, and recovery code. The integrity of this level is assumed to be valid. We do, however, perform an initial checksum test in order to identify PROM failures. The first level contains the remainder of the usual BIOS code. The second level contains all of the expansion cards and their associated ROMs, if any. The third level contains the operating system boot block(s). These are resident on the bootable device and are responsible for loading the operating system kernel. The fourth level contains the operating system, and the fifth and final level contains user level programs and any network hosts.”)
1-A	selecting a program residing in the volatile memory,	p. 66 (“To have a practical impact, AEGIS must be able to work with commodity hardware with minimal changes (ideally none) to the existing architecture. The IBM PC architecture was selected as our prototype platform because of its large user community and the availability of the source code for several operating systems. . . . AEGIS modifies the boot process shown in figure 2 so that all executable code,

Exhibit 7

Claim Chart for U.S. Patent No. 6,411,941 based on DMI Specification (Desktop Management BIOS Specification)

Element	Claim Language	DMI Specification
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	To the extent that it is found that the DMI Specification does not read on this claim element, this claim element would have been obvious to one of ordinary skill in the art, based on the factors enumerated in <i>KSR Int'l Co. v. Teleflex Inc.</i> , 550 U.S. 398, 419-421 (2007) (“For the same reason, if a technique has been used to improve one device, and a person of ordinary skill in the art would recognize that it would improve similar devices in the same way, using the technique is obvious unless its actual application is beyond his or her skill” and “[a] court can take account of the inferences and creative steps that a person of ordinary skill in the art would employ” in determining obviousness) and the DMI Specification’s combination with other prior, including but not limited to the following prior art: Yee 1995, Tygar 1991, Arbaugh 1996, Hannah Patent, Misra Patent, Joshi Patent, Miller Patent, Lewis Patent, Clark Patent, Chou Patent, or Davis 692 Patent.
1-A	selecting a program residing in the volatile memory,	To the extent that it is found that the DMI Specification does not read on this claim element, this claim element would have been obvious to one of ordinary skill in the art, based on the factors enumerated in <i>KSR Int'l Co. v. Teleflex Inc.</i> , 550 U.S. 398, 419-421 (2007) (“For the same reason, if a technique has been used to improve one device, and a person of ordinary skill in the art would recognize that it would improve similar devices in the same way, using the technique is obvious unless its actual application is beyond his or her skill” and “[a] court can take account of the inferences and creative steps that a person of ordinary skill in the art would employ” in determining obviousness) and the DMI Specification’s combination with other prior, including but not limited to the following prior art: Yee 1995, Tygar 1991, Arbaugh 1996, Hannah Patent, Misra Patent, Joshi Patent, Miller Patent, Lewis Patent, Clark Patent, Chou Patent, or Davis 692 Patent.

Exhibit 8**Claim Chart for U.S. Patent No. 6,411,941 based on Arbaugh, A Secure and Reliable Bootstrap Architecture**

Element	Claim Language	Arbaugh II
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	p. 1 (“We have designed AEGIS, a secure bootstrap process. AEGIS increases the security of the boot process by ensuring the integrity of bootstrap code. It does this by constructing a chain of integrity checks, beginning at power-on and continuing until the final transfer of control from the bootstrap components to the operating system itself. The integrity checks compare a computed cryptographic hash value with a stored digital signature associated with each component.”) p. 4 (“We have divided the boot process into several levels to simplify and organize the AEGIS BIOS modifications, as shown in figure 3. Each increasing level adds functionality to the system, providing correspondingly higher levels of abstraction. The lowest level is Level 0. Level 0 contains the small section of trusted software, digital signatures, public key certificates, and recovery code. The integrity of this level is assumed to be valid. We do, however, perform an initial checksum test in order to identify PROM failures. The first level contains the remainder of the usual BIOS code. The second level contains all of the expansion cards and their associated ROMs, if any. The third level contains the operating system boot block(s). These are resident on the bootable device and are responsible for loading the operating system kernel. The fourth level contains the operating system, and the fifth and final level contains user level programs and any network hosts.”)
1-A	selecting a program residing in the volatile memory,	p. 2-3 (“To have a practical impact, AEGIS must be able to work with commodity hardware with minimal changes (ideally none) to the existing architecture. The IBM PC architecture was selected as our prototype platform because of its large user community and the availability of the source code for several operating systems. . . .

Exhibit 9**Claim Chart for U.S. Patent No. 6,411,941 based on Hellman (U.S. Patent No. 4,658,093)**

Element	Claim Language	Hellman
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	1:4-7 ("The invention relates to a software distribution system and more particularly to a secure software distribution system in which the number of uses of software can be controlled.") 4:22-27 ("Accordingly, it is an objective of the invention to allow a software manufacturer to control the number of times a piece of software is used by a customer, in a manner such that the authorization cannot be recorded and reused, and such that the authorization is not transferable to another base unit.") 4:34-36 ("Another objective of the invention is to prevent "software piracy." That is the illegal use of software on a base unit which has not paid a license fee.")
1-A	selecting a program residing in the volatile memory,	5:51-53 ("The user at base unit 12 obtains software package 17 by purchasing it at a store, over telephone line, or in some similar manner.") 6:48-59 ("As with any has function, the one-way has function generator 22 has a much shorter output, perhaps 100 bits, than its input, typically 10,000 to 1,000,000 bits in this invention. This allows H to be stored more economically than the software package 17 in the base unit's memory. Storage of H or of the complete software package 17 is desirable to increase the security of the system because storage of only SOFTWARE NAME would allow a dishonest user to rename software packages and pay for uses of less expensive packages when really using more expensive ones.")

Exhibit 10**Claim Chart for U.S. Patent No. 6,411,941 based on Joshi (U.S. Patent No. 4,688,169)**

Element	Claim Language	Joshi
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	1:5-8 (“This invention relates to a computer software security system, and more particularly to a security system which prevents unauthorized execution of a program.”) 2:25-29 (“It is therefore an object of this invention to provide an improved computer software security system which prevents a program from executing on computers other than an authorized machine.”) 2:65-69 (“In one embodiment, the first storage means is a memory circuit which may include a storage circuit printed on a circuit board in the machine, or may include a programmable read-only memory.”) 4:28-32 (“The unique machine identification code may be located in a hardware or firmware memory circuit such as a storage circuit printed on a circuit board in the machine, or such as a programmable read-only memory.”) 4:39-42 (“The operating system typically obtains the unique machine identification code from one or more of the storage areas in the computer hardware or firmware.”) 5:5-10 (“A block diagram of a typical computer system utilizing security system 10 is shown in FIG. 2. Central processing unit 26 is responsive to keyboard input 28, disk input/output 30, random access memory 32, programmable read-only memory circuit 34, and other peripherals 36 including serial I/O.”) Claim 20 (“20. A method for restricting execution of a computer software program to a particular machine comprising: . . . ”)

Exhibit 11**Claim Chart for U.S. Patent No. 6,411,941 based on Allen (U.S. Patent No. 4,757,533)**

Element	Claim Language	Allen
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	Abstract (“A security system for a personal computer, in which hardware and software are combined to provide a tamper-proof manner of protecting user-access and file-access.”) 1:49-54 (“It is, therefore, the primary objective of the present invention to provide a personal computer [sic] security system that prevents access to the operating system of the personal computer by invalid users, and also protects against access to selected files of the computer on a user-by-user basis.”) 2:28-40 (“[T]he security sy[s]tem of the present invention includes as its hardware portion an expansion board insertable into one of the expansion slots of the personal computer. The expansion board includes an EPROM on which is stored object code. There are four portions to the EPROM: the first portion contains code for boot-process handling; the second portion contains code for the main file handling interrupt function calls of the operating system; the third portion contains code for the available interrupt function calls of the present invention; and the fourth portion contains object code for the monitoring of the hard disc format interrupt vector.”) 2:41-45 (“The expansion board of the present invention also includes . . . an erasable nonvolatile memory, such as EEPROM, on which are stored the passwords of the valid users of the personal computer”)
1-A	selecting a program residing in the volatile memory,	23:47-49 (“FIG. 14 is a flow chart for the DOS function call ‘Load or Execute a Program’, DOS function call 4BH (for DOS 2.00 and above).”) FIG. 14

Exhibit 12

Claim Chart for U.S. Patent No. 6,411,941 based on Karp (U.S. Patent No. 4,866,769)

Element	Claim Language	Karp
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	1:6-10 (“This invention relates to copy protection of software on publicly-distributed diskettes through the use of an encryption technique which authenticates a uniquely-identified computing system as the one on which the software has originally been installed.”) 3:8-11 (“The invention is expressed as a method of controlling the use and replication of diskette software contents and the like on unauthorized diskette-driven computing systems.”) 4:26-28 (“The invention is executable upon a combination including a personal computer (PC) 10 and a magnetic disk (“diskette”) 12.”)
1-A	selecting a program residing in the volatile memory,	4:54-63 (“The application program on the diskette is transferred in whole or in part from the diskette to the primary memory of the personal computer where it is used to execute an application for the user of the computer. For example, a word processing program on the diskette 12 can be loaded through the disk driver 18 into the primary memory (not shown) of the personal computer 10, where it can support the production and processing of documents created by the user of the PC 10.”) 5:46-53 (“The INITIALIZE module 22 may include interaction with the user, but certainly involves interaction with the operating system 16 for the purpose of transferring the application program 24 in whole or in part into the primary memory of the PC 10 and setting initial constant values. Once the application program 24 is installed and initialized, it can be executed on the PC 10.”) 7:62-8:1 (“As shown in FIG. 3 and Table II, when the software 20 is initially installed in the PC 10, the INSTALL module is called to establish the parameters in configuration of the PC 10 that are determinative of the mode of execution of the

Exhibit 13**Claim Chart for U.S. Patent No. 6,411,941 based on Waite et al. (U.S. Patent No. 5,103,476)**

Element	Claim Language	Waite et al.
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	1:28-31 (“Furthermore, a method and system in which the licensor is afforded protection as well as more precisely monitoring the licensees is warranted.”) 2:27-30 (“The purpose of the present invention is to allow a licensor to maintain accountability of its programs in a manner much more efficient than presently being utilized.”)
1-A	selecting a program residing in the volatile memory,	1:66-2:6 (“Once all of the appropriate information is relayed to the registration database computer, the essential segments of the particular program are transferred or transmitted as a tamperproof overlay file to the registration shell and installed on the personal computer. Subsequently, the loader segment will activate the main program by providing the essential segment each time the main program is loaded for execution.”) 2:36-39 (“Initially, a particular program which does not contain a critical or essential segment is provided in a personal computer or other device on a magnetic disc, firmware, hardware, or other means.”) 2:66-3:1 (“The registration shell program is executed when the user first attempts to execute the product application program provided in the main program file 16 of the PC operating system 14.”) 3:65-4:7 (“Upon receipt of the complete shipping file, an unpacker/installer subprogram 28 in the registration shell program opens the shipping file and installs the tamperproof overlay file 40 including the critical program segment 24, CRC value 22, as well as the decryption key 20 and the updated main program files, if

Exhibit 14**Claim Chart for U.S. Patent No. 6,411,941 based on Waite et al. (U.S. Patent No. 5,222,134)**

Element	Claim Language	Waite et al.
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	2:3-5 (“Consequently, an economical method for distributing software to a mass market while protecting the software from abuse by anonymous licensees is warranted.”) 2:13-16 (“The present invention is directed to a method and a system for permitting personal computer software programs or other types of programs to be distributed in a license controlled manner.”) 3:30-33 (“The purpose of the present invention is to allow a licensor to maintain accountability of its programs in a manner much more efficient than presently being utilized.”)
1-A	selecting a program residing in the volatile memory,	2:60-68 (“A unique encrypted package is assembled containing some data unique to the licensed user and an operational version of the program combined within an encrypted file. A unique decryption key is transmitted to the user's computer along with the encrypted file and any unprotected program files which may augment the marketing shell. Upon receiving the decryption key, encrypted file, and unprotected files, the marketing shell will install each of these on the user's computer.”) 3:1-13 (“Subsequently, each time the user executes the program, a loader segment will load and decrypt the encrypted file, as an overlay to the unprotected files, using the decryption key provided. The program will be executed in accordance with the design of the protected software program and the unique licensee data is also loaded during the program execution. When the program is not executing, the protected program remains in its encrypted form, in the mass storage of the computer along with the unprotected program files. The protected program is decrypted only when loaded for execution and may not be changed without access to the authentic encryption key.”)

sf- 3061724

1

Exhibit 15**Claim Chart for U.S. Patent No. 6,411,941 based on Smyth (5,325,430)**

Element	Claim Language	Smyth
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	Fig. 1 4:37-39 (“The public RAM 11 here provides for 32 Kbytes of storage and is used for specific variable storage and provides a sector buffer for encryption/decryption.”) 5:37-39 (“The standard BIOS ROMs are placed with a modified set of BIOS ROMs including the BIOS patch 68 (FIG. 2).”) 7:25-35 (“In accordance with the present invention, a variety of patches have been added to the BIOS. The significant modifications to the BIOS are: . . . Intercept to security module startup code;”)
1-A	selecting a program residing in the volatile memory,	6:20-25 (“Turning to the other aspects of the security system, principally the software aspects, these will now be described in relation to FIG. 2. Here again the hardware of the security module is indicated at 1, and the basic computer lap top is indicated at 33. Conventional application software is indicated at 60, and this communicates with DOS indicated at 62 and BIOS indicated at 64. In known manner, the application software can then communicate directly with the hardware through BIOS 64 or indirectly through the DOS 62 and then BIOS 64.”) 13:20-22 (“The user will then be given the option of accessing various applications, e.g. word processing applications, spread sheets and data processing programs.”)
1-B	using an agent to set up a verification structure in the erasable, non-volatile memory of	14:7-11 (“The first, low level is data stored on disc used to ‘boot strap’ the computing device. In known manner, this is the program which loads DOS into the

Exhibit 16**Claim Chart for U.S. Patent No. 6,411,941 based on Ewertz (U.S. Patent No. 5,371,876)**

Element	Claim Language	Ewertz
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	Abstract (“A computer system wherein a paging technique is used to expand the usable non-volatile memory capacity beyond a fixed address space limitation. The computer system of the preferred embodiment includes a flash memory component for storing non-volatile code and data including a system BIOS in the upper 128K of memory. The useful BIOS memory space is effectively increased while maintaining the address boundary of the upper 128K region. The address space of the non-volatile memory device is logically separated into distinct pages of memory (Pages 1-4). Page 1, Page 3 and Page 4 may be individually swapped into the address space originally occupied by Page 1 (the swappable page area). In the preferred embodiment, Page 2 is held static and thus is not used as a swap area. Each of the swappable pages contain processing logic called swapping logic used during the swapping or paging operation. The swapping logic operates in conjunction with paging hardware to effect the swapping of pages into the swappable page area. The high order processor address lines are input by a page decoder. The page decoder is used to modify the address actually presented to the non-volatile memory device. A page register is used by the processor to select a page in non-volatile memory. In an alternative embodiment, several different forms of configuration or identification information may be stored in a page of non-volatile memory.”) 1:56-58 (“These resources include ROM, random access memory (RAM), input/output devices, and possibly other processors.”) 3:27-29 (“Identification information may include an Ethernet address, system serial numbers, or software license numbers.”) 5:1-3 (“The flash memory device used in the preferred embodiment contains four

Exhibit 17**Claim Chart for U.S. Patent No. 6,411,941 based on Davis (U.S. Patent No. 5,473,692)**

Element	Claim Language	Davis
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	Abstract (“An integrated circuit component for enforcing licensing restrictions. Such enforcement is performed through remote transmission of access privileges for executing a licensed program from the integrated circuit component to another similar component. The integrated circuit component comprising a non-volatile memory for storing a uniquely designated key pair, an authentication device certificate and a manufacturer public key along with cryptographic algorithms, a processor for executing the cryptographic algorithms in order to process information inputted into the integrated circuit component and for transmitting the processed information into volatile memory and a random number generator for generating the uniquely designated key pair internally within the integrated circuit component.”) 1:16-22 (“The present invention relates to licensing software. More particularly, the present invention relates to an apparatus and method for transferring access privileges for executing a licensed software program from an authorized node having a first hardware agent to a non-authorized node having a second hardware agent without violating a specific-user license.”) 5:63-6:4 (“Referring to FIG. 4, an illustrative embodiment of a computer system 100 utilizing the present invention is illustrated. The computer system 100 comprises a plurality of bus agents including a host processor 105, a memory device 110, an input/output (“I/O”) controller 115 and a cryptographic device referred to as a “hardware agent” 120. The plurality of bus agents are coupled together through a system bus 130 which enables information to be communicated between these bus agents.”)

Exhibit 18**Claim Chart for U.S. Patent No. 6,411,941 based on Richardson (U.S. Patent No. 5,490,216)**

Element	Claim Language	Richardson
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	8:48-56 ("The distinction as against the first embodiment is that a "key file" is created at the time of registration of the software and a duplicate key file is also created at the same time. The duplicate key file is arranged to be stored on the computer at a location separate from the program to be protected. In the case of the Apple Macintosh computer the duplicate key file can be stored in the "system" folder.")
1-A	selecting a program residing in the volatile memory,	5:47-56 ("With reference to FIGS. 1 and 8, the system according to embodiments of the invention is designed and adapted to allow digital data 39 or software to run in a use mode on a platform 31 if and only if an appropriate licensing procedure has been followed. In particular forms, the system includes means for detecting when parts of the platform 31 on which the digital data 39 has been loaded has changed in part or in entirety as compared with the platform parameters when the software or digital data to be protected was, for example, last booted or run or validly registered.") 6:30-33 ("Included as part of the software on that disk 10 is a registration and re-registration routine which executes whenever the program protected by the arrangement of the first embodiment "boots".") 8:6-14 ("The registration routine behaves generally as follows where any copy of the protected software boots. In this situation, the registration routine checks at the time of boot to see what registration details are present for that particular copy of the software. If no details are present, then it is assumed that the PC is booting from a newly distributed magnetic disk and registration is to occur for the first time. The registration procedure in that case is that followed in respect of FIGS. 2a, 2b and

Exhibit 19**Claim Chart for U.S. Patent No. 6,411,941 based on Schull (U.S. Patent No. 5,509,070)**

Element	Claim Language	Schull
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	8:31-50 ("In this paragraph we disclose a software-based method we have developed to retrofit Target-IDs with writeable non-volatile storage devices (such as computer hard drives or programmable read-only-memory (PROM) chips. By this method, a nearly-unique "tattoo" is generated and written to the storage device. The near-uniqueness of the "tattoo" can be achieved by using a random number generator initialized or "seeded" in a fashion which will vary from one ID-Target to the next (this facility is typically incorporated into a "randomize" command in many programming languages). Alternatively or equivalently, a nearly-unique ID can be based upon a high-resolution time-stamp generated using the processor's system clock, or by a variety of other means. To the extent that this "tattoo" is the only source of the Target-ID's uniqueness it is desirable that it not be copyable from one ID-target to another. This can be achieved by hiding the tattoo from the User, copy-protecting it, etc., using techniques known to those with ordinary skill in the prior art.") 16:32-49 ("In a non-volatile storage location of the User's Processor such as the User's hard drive, store a "profile" which records an encrypted list of the N characteristics of the ID-target on which the Target-ID is based. Create or update this file whenever a valid password is encountered. Whenever an invalid password is encountered compare the values stored within the profile to the values of the Target-ID which has just been computed. In the latter case, construct an N bit word, giving each bit a value (1 or 0) which indicates whether or not the Nth characteristic of the profile differs from the Nth characteristic responsible for the Target ID. This N bit word, transformed or encrypted as desired, can then be communicated to the Licensing System's Customer Service operator who determine which characteristics changed since the last time a valid password was installed on the User's Processor. This determination can then be used check the User's veracity,

Exhibit 20**Claim Chart for U.S. Patent No. 6,411,941 based on Morisawa (U.S. Patent No. 5,537,544)**

Element	Claim Language	Morisawa
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	6:60-62 (“The password memory means 2A is realized by an EEPROM 29 coupled to the keyboard controller (KBC) 30 through the PCMCIA gate array (PCMCIA-GA) 28.”) 7:34-45 (“When the registration of the passwords (the number of registered passwords=)/(is verified by a response content (the number of registered passwords) from the keyboard controller (KBC) 30 upon issuing of the command, password input message data is sent to a VGA controller 32 by the BIOS calling to display a password input requesting message on an LCD panel 49, ...”) 10:15-20 (“The BIOS-ROM 25 stores a Basic Input and Output System program and formed of a flash memory (FLASH MEM) so that the BIOS program can be modified. . . . The flash memory (FLASH MEM) serving as the BIOS ROM 25 can be initialized only when a maintenance privilege is set as an access right in the security register of the PCMCIA gate array (PCMCIA-GA) 28, and the control will be described later.”) 3:4-23 (“According to a third aspect of the present invention, there is provided a portable computer system which comprises . . . a password control section, holding one or more registered passwords as being unreadable by direct access from the main processor section, for allowing the main processor section to perform the data processing when a password identical to one of the registered passwords held therein is input by the input section, wherein the password control section includes a non-volatile password memory for storing one or more registered passwords, and a password checking section for checking whether the input password coincides with any one of the registered passwords stored in the password memory, the main processor section is connected to the password checking section via a system bus,

Exhibit 21**Claim Chart for U.S. Patent No. 6,411,941 based on Davis (U.S. Patent No. 5,568,552)**

Element	Claim Language	Davis
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	Abstract (“An integrated circuit component for enforcing licensing restrictions. Such enforcement is performed through remote transmission of access privileges for executing a licensed program from the integrated circuit component to another similar component. The integrated circuit component comprising a non-volatile memory for storing a uniquely designated key pair, an authentication device certificate and a manufacturer public key along with cryptographic algorithms, a processor for executing the cryptographic algorithms in order to process information inputted into the integrated circuit component and for transmitting the processed information into volatile memory and a random number generator for generating the uniquely designated key pair internally within the integrated circuit component.”) 1:19-25 (“The present invention relates to licensing software. More particularly, the present invention relates to an apparatus and method for transferring access privileges for executing a licensed software program from an authorized node having a first hardware agent to a non-authorized node having a second hardware agent without violating a specific-user license.”) 5:50-58 (“Referring to FIG. 4, an illustrative embodiment of a computer system 100 utilizing the present invention is illustrated. The computer system 100 comprises a plurality of bus agents including a host processor 105, a memory device 110, an input/output (“I/O”) controller 115 and a cryptographic device referred to as a “hardware agent” 120. The plurality of bus agents are coupled together through a system bus 130 which enables information to be communicated between these bus agents.”)

Exhibit 22

Claim Chart for U.S. Patent No. 6,411,941 based on Christenson et al. (U.S. Patent No. 5,579,522)

Element	Claim Language	Christenson et al.
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	To the extent that it is found that Christensen does not read on this claim element, this claim element would have been obvious to one of ordinary skill in the art, based on the factors enumerated in <i>KSR Int'l Co. v. Teleflex Inc.</i> , 550 U.S. 398, 419-421 (2007) (“For the same reason, if a technique has been used to improve one device, and a person of ordinary skill in the art would recognize that it would improve similar devices in the same way, using the technique is obvious unless its actual application is beyond his or her skill” and “[a] court can take account of the inferences and creative steps that a person of ordinary skill in the art would employ” in determining obviousness) and Christensen’s combination with other prior, including but not limited to the following prior art: Yee 1995, Tygar 1991, Arbaugh 1996, Hannah Patent, Joshi Patent, Miller Patent, Lewis Patent, Clark Patent, Chou Patent, or Davis 692 Patent.
1-A	selecting a program residing in the volatile memory,	To the extent that it is found that Christensen does not read on this claim element, this claim element would have been obvious to one of ordinary skill in the art, based on the factors enumerated in <i>KSR Int'l Co. v. Teleflex Inc.</i> , 550 U.S. 398, 419-421 (2007) (“For the same reason, if a technique has been used to improve one device, and a person of ordinary skill in the art would recognize that it would improve similar devices in the same way, using the technique is obvious unless its actual application is beyond his or her skill” and “[a] court can take account of the inferences and creative steps that a person of ordinary skill in the art would employ” in determining obviousness) and Christensen’s combination with other prior, including but not limited to the following prior art: Yee 1995, Tygar 1991, Arbaugh 1996, Hannah Patent, Joshi Patent, Miller Patent, Lewis Patent, Clark Patent, Chou Patent, or Davis 692 Patent.

Exhibit 23**Claim Chart for U.S. Patent No. 6,411,941 based on McCarty (U.S. Patent No. 5,666,411)**

Element	Claim Language	McCarty
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	Abstract (“This system protects proprietary software from disclosure and unauthorized use, enforces license limits on number of users of the software, and prevents corruption of protected software by computer viruses. Software protected under this system may execute only on computer systems which incorporate a microprocessor capable of deciphering enciphered instructions in real time. Program files are first enciphered under control of a distribution cipher key. Prior to first use of software, program files must be customized on the user computer system. This customization procedure re-enciphers the programs, so that they are enciphered under a second cipher key. Customized programs may not execute on a computer system other than one constructed with a processor chip which incorporates a crypto microprocessor. The crypto microprocessor is capable of performing this re-encipherment, and of executing both enciphered and unenciphered programs. The customization program runs on user's computer system and normally accesses a remote Exchange database system by means of a modem to accomplish its task. Variations of customization process provide for storage of enciphered software on either a single system, a network server, or a site license repository system.”) 1:6-8 (“This invention relates to a system for protecting computer software. More specifically, this invention relates to a cryptographic system for protecting mass marketed software.”) 4:24-34 (“Designed to prevent software piracy, the present invention comprises a method of enciphering computer programs, a method of customizing the programs by the user, an improved crypto processor, and a method of execution in a computer system incorporating this crypto processor. Accordingly, several objects and advantages of the present invention are:

Exhibit 24**Claim Chart for U.S. Patent No. 6,411,941 based on Lewis (U.S. Patent No. 5,734,819)**

Element	Claim Language	Lewis
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	Abstract (“A method and apparatus for providing system operation validation is disclosed. The method and apparatus for validation operates within a computer system comprising a central processing unit coupled to a programmable memory, and to a system device. The programmable memory may store programs and instructions executable on the CPU and a non-volatile memory is also provided for access by the CPU. The system operation validation is provided by a chip identifier located within a device memory within the system device, which memory also serves as a chip identifier register. Selected information stored within the non-volatile memory is used, along with the chip identifier, to generate a first encryption code associated with the system device. An encryption key is used to generate a second encryption code associated with the computer system. The first and second encryption codes are matched to provide a first level system operation validation. A second chip identifier is generated, which identifier is associated with the computer system. Both chip identifiers are compared to provide a second level system operation validation.”) 1:8-16 (“The present invention relates, generally, to a computer system having a non-volatile memory and, more specifically, to electronic security information being stored in the non-volatile memory. More specifically, the present invention relates to a computer system having a non-volatile memory with security information written into the non-volatile memory and a way of detecting when that information has been altered so as to prevent operation of the computer system once tampering has been detected.”) 1:31-37 (“To provide the manufacturer the flexibility to write a different serial number on each machine, the machine is designed with some type of non-volatile memory (NVM) that the manufacturer can write (such as EPROM, PROM, ROS,

Exhibit 25**Claim Chart for U.S. Patent No. 6,411,941 based on O'Connor (U.S. Patent No. 5,745,568)**

Element	Claim Language	O'Connor
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	1:6-9 (“This invention relates to the field of computer system manufacturing processes. More specifically, this invention relates to a process for configuring and restricting a computer system software to a specific computer hardware.”) 3:34-49 (“In one implementation, an identifier is permanently recorded in a non-volatile memory present in the hardware and accessible using a BIOS routine. In another implementation, the identifier is permanently programmed into a segment or multiple segments of a BIOS memory which semi-uniquely identifies a computer system as a member of a group such as a class of computer systems or computer systems produced by a particular manufacturer. One technique for converting a segment or group of segments to a hardware identifier is accomplished using a BIOS routine that performs a checksum operation, as is well known in the art of computing, on the segment or segment group. Alternatively, a BIOS routine converts a segment or group of segments to an identifier by applying a data compression scheme, as is well known in the art of computing, to the segment or segment group.”) 4:5-13 (“A verify software-hardware association step 134 reads the hardware identifier written to a loaded CD-ROM, compares the hardware identifier read from the CD-ROM to the hardware identifier retrieved from the computer system hardware and terminates the CD-ROM access program 130 if the hardware identifiers do not match. Otherwise, in decrypt software files routine 136, the hardware identifier is then used as a decryption key to decrypt the software program files.”) 6:10-16 (“The CD-ROM access program compares the computer hardware identifier code written to the CD-ROM against the computer hardware identifier

sf-3054975

1

Exhibit 26**Claim Chart for U.S. Patent No. 6,411,941 based on Davis (U.S. Patent No. 5,844,986)**

Element	Claim Language	Davis
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	Abstract (“A subsystem prevents unauthorized modification of BIOS program code embedded in modifiable non-volatile memory devices such as flash memory. A cryptographic coprocessor containing the BIOS memory device performs authentication and validation on the BIOS upgrade based on a public/private key protocol. The authentication is performed by verifying the digital signature embedded in the BIOS upgrade.”) 1:6-9 (“This invention relates to the field of security of computer firmware, especially in the areas of Basic Input and Output System ("BIOS") in general computing systems, such as personal computers ("PCs").”) 1:11-19 (“One of the most critical elements in a computer system is the boot-up firmware, such as the Basic Input and Output System ("BIOS"). Typically stored in some form of non-volatile memory, the BIOS is machine code, usually part of an Operating System ("OS"), which allows the Central Processing Unit ("CPU") to perform tasks such as initialization, diagnostics, loading the operating system kernel from mass storage, and routine input/output ("I/O") functions.”) 1:63-67 (“The primary focus of the present invention, therefore, is to prevent corrupting the BIOS by a computer virus. This is achieved by imposing an authentication and validation procedure before the contents of the BIOS flash memory are modified.”) 2:64-3:9 (“Referring to FIG. 1, an embodiment of a computer system implemented within the present invention is shown. The computer system 10 includes a chipset 31 which operates as an interface to support communications between host

Exhibit 27

Claim Chart for U.S. Patent No. 6,411,941 based on Clark (U.S. Patent No. 5,892,902)

Element	Claim Language	Clark
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	FIG. 3 3:24-44 (“As depicted in FIG. 3, the intelligent token 10 stores a protected copy of the file that is usually stored in a disk boot sector of a computer along with other file integrity data The EEPROM 18 preferably stores software programs to enforce access control to the host computer and access control to the remote domain 50 (access control software 20). . . . In addition, critical information such as boot sector information 22, file integrity information 24 and authentication information 26 may be stored in the EEPROM 18. The authentication information may include, e.g., file signature information and cryptographic keys for both the host and remote computers. . . . Traditionally, the above described critical information has been stored in the host computer’s boot sector memory.”)
1-A	selecting a program residing in the volatile memory,	4:31-53 (“As explained in U.S. Pat. No. 5,448,045, during system startup, two authentications must be successfully performed to complete the boot sequence If both the user and card authentication are successful, the boot sequence completes, and control is given to the host computer operating system – some or all of which has been retrieved from the intelligent token 10. The user may then proceed to utilize the host computer in the usual fashion, uploading additional information, i.e., applications or application integrity information from the intelligent token 0 as needed.”)
1-B	using an agent to set up a verification structure in the erasable, non-volatile memory of the BIOS, the verification structure accommodating data	FIG. 3, 5, 6, 7 4:31-53 (“As explained in U.S. Pat. No. 5,448,045, during system startup, two authentications must be successfully performed to complete the boot sequence If both the user and card authentication are successful, the boot sequence

Exhibit 28**Claim Chart for U.S. Patent No. 6,411,941 based on Chou (5,892,906)**

Element	Claim Language	Chou
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	1:54-2:7 (“Many computer manufacturers have implemented password protection in the computer BIOS (Basic Input/Output System) which is integral to the operation of a personal computer. The password protection in the BIOS halts the system boot up unless the system boot up unless the user enters a password which is also stored in the foregoing CMOS RAM. . . . Recent changes in the computer BIOS memory storage devise permit writing data to the BIOS memory, offering the opportunity to provide password protection within the same memory which stores the BIOS routines. Thus, any attempt to delete the protection will result in the BIOS routine being disabled, disabling the boot up process. EEPROM flash deices may be programmed with BIOS routines which permit the user to enter data without requiring the computer to be returned to the manufacture. The present invention makes use of these new BIOS memory devices for effecting security measures which discourage theft.”) 2:14-23 (“As a minimum, the security routine requires verification of a password entered by the user, or a verification of a quantity read from an externally connected memory device. The security function stored in the BIOS memory also includes an administration function which may be invoked by the user following the normal boot up sequence initiated by the BIOS once the password or validation quantity is verified.”) 2:41-47 (“In another embodiment of the invention, the locked state requires the user to manually enter the password through the keyboard in response to a prompt during execution of the BIOS routine. The security function compares a unique, user defined password stored in the BIOS memory to the user supplied password. If the two passwords agree, the computer completes execution of the BIOS routine.”)

Exhibit 29

Claim Chart for U.S. Patent No. 6,411,941 based on Labatte et al. (U.S. Patent No. 5,901,311)

Element	Claim Language	Labatte et al.
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	To the extent that it is found that Labatte does not read on this claim element, this claim element would have been obvious to one of ordinary skill in the art, based on the factors enumerated in <i>KSR Int'l Co. v. Teleflex Inc.</i> , 550 U.S. 398, 419-421 (2007) (“For the same reason, if a technique has been used to improve one device, and a person of ordinary skill in the art would recognize that it would improve similar devices in the same way, using the technique is obvious unless its actual application is beyond his or her skill” and “[a] court can take account of the inferences and creative steps that a person of ordinary skill in the art would employ” in determining obviousness) and Labatte’s combination with other prior, including but not limited to the following prior art: Yee 1995, Tygar 1991, Arbaugh 1996, Hannah Patent, Joshi Patent, Miller Patent, Lewis Patent, Clark Patent, Chou Patent, or Davis 692 Patent.
1-A	selecting a program residing in the volatile memory,	To the extent that it is found that Labatte does not read on this claim element, this claim element would have been obvious to one of ordinary skill in the art, based on the factors enumerated in <i>KSR Int'l Co. v. Teleflex Inc.</i> , 550 U.S. 398, 419-421 (2007) (“For the same reason, if a technique has been used to improve one device, and a person of ordinary skill in the art would recognize that it would improve similar devices in the same way, using the technique is obvious unless its actual application is beyond his or her skill” and “[a] court can take account of the inferences and creative steps that a person of ordinary skill in the art would employ” in determining obviousness) and Labatte’s combination with other prior, including but not limited to the following prior art: Yee 1995, Tygar 1991, Arbaugh 1996, Hannah Patent, Joshi Patent, Miller Patent, Lewis Patent, Clark Patent, Chou Patent, or Davis 692 Patent.

Exhibit 30**Claim Chart for U.S. Patent No. 6,411,941 based on Labatte et al. (U.S. Patent No. 5,913,057)**

Element	Claim Language	Labatte et al.
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	To the extent that it is found that Labatte does not read on this claim element, this claim element would have been obvious to one of ordinary skill in the art, based on the factors enumerated in <i>KSR Int'l Co. v. Teleflex Inc.</i> , 550 U.S. 398, 419-421 (2007) (“For the same reason, if a technique has been used to improve one device, and a person of ordinary skill in the art would recognize that it would improve similar devices in the same way, using the technique is obvious unless its actual application is beyond his or her skill” and “[a] court can take account of the inferences and creative steps that a person of ordinary skill in the art would employ” in determining obviousness) and Labatte’s combination with other prior, including but not limited to the following prior art: Yee 1995, Tygar 1991, Arbaugh 1996, Hannah Patent, Joshi Patent, Miller Patent, Lewis Patent, Clark Patent, Chou Patent, or Davis 692 Patent.
1-A	selecting a program residing in the volatile memory,	To the extent that it is found that Labatte does not read on this claim element, this claim element would have been obvious to one of ordinary skill in the art, based on the factors enumerated in <i>KSR Int'l Co. v. Teleflex Inc.</i> , 550 U.S. 398, 419-421 (2007) (“For the same reason, if a technique has been used to improve one device, and a person of ordinary skill in the art would recognize that it would improve similar devices in the same way, using the technique is obvious unless its actual application is beyond his or her skill” and “[a] court can take account of the inferences and creative steps that a person of ordinary skill in the art would employ” in determining obviousness) and Labatte’s combination with other prior, including but not limited to the following prior art: Yee 1995, Tygar 1991, Arbaugh 1996, Hannah Patent, Joshi Patent, Miller Patent, Lewis Patent, Clark Patent, Chou Patent, or Davis 692 Patent.

Exhibit 31**Claim Chart for U.S. Patent No. 6,411,941 based on Griswold (U.S. Patent No. 5,940,504)**

Element	Claim Language	Griswold
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	Abstract (“A license management system and method for recording the use of a licensed product, and for controlling its use in accordance with the terms of the license. A licensed product invokes a license check monitor at regular time intervals. The monitor generates request datagrams which identify the licensee and the product and sends the request datagrams over a communications facility to a license control system. The license control system maintains a record of the received datagrams, and compares the received datagrams to data stored in its licensee database. Consequently, the license control system transmits reply datagrams with either a denial or an approval message to the monitor. The monitor terminates further use of the product if it receives a denial message. The monitor generates its own denial message if its request datagrams are unanswered after a predetermined interval of time. The datagrams are counted at the control system to provide billing information.”) 1:16-22 (“The present invention generally relates to systems for managing licenses of products such as computer software, video games, CD-ROM information, movies and other video products, music and other audio products, multimedia products, and other systems for up-to-date recording of actual usage of such a licensed product to enable efficient billing therefor.”) 3:31-34 (“It is an object of the present invention to provide a license management system and method which can ensure that a licensed product is used only on machines under which it is licensed.”)
1-A	selecting a program residing in the volatile memory,	5:19-23 (“As shown in FIG. 1, a licensed product 1 is located at a licensee's site. Product 1 may include a data portion 1B and a functional portion 1A such as computer software product or any other kind of information product used to control use of data portion 1B.”)

Exhibit 32

Claim Chart for U.S. Patent No. 6,411,941 based on Beelitz (U.S. Patent No. 5,944,820)

Element	Claim Language	Beelitz
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	Abstract (“A system and method for providing a modifiable partition boot record for a memory device of a computer. The computer includes a processor for loading and running boot code and operating code. The memory device, which can be accessed by the processor, includes at least two partitions, the first partition initially being active and having a type that is not recognized by the operating system code and the second partition initially being inactive. When the computer boots up, the processor first accesses the active partition. Once the active partition has been accessed, the processor then converts the type for the active partition to a visible type, i.e, a type that is recognizable to the operating system code. When the operating system code is loaded, it recognizes the type for the active partition and therefore does not encounter an error. Once the operating system code has been loaded, the processor converts the type for the active partition back to a non-visible type. Because the operating system is now loaded, the processor can perform other tasks. For example, the processor can perform an EBTS program that prompts a user for acceptance of a software license, the software being stored in the second partition. If the user accepts the software license, the processor then designates the second partition as active.”) 1:5-2:24 (“The present disclosure relates generally to computers and more specifically to a partition boot record for a computer memory device that can change types dynamically, such as from a non-recognizable boot record to a recognizable boot record. In a computer having one or more storage devices such as a hard disk drive, the storage devices are often separated into different partitions so that certain data or programs can be stored on separate partitions. For example, in a computer that utilizes a Microsoft DOS, Windows, or Windows 95 operating system, as provided

Exhibit 33**Claim Chart for U.S. Patent No. 6,411,941 based on Okada et al. (U.S. Patent No. 6,049,670)**

Element	Claim Language	Okada et al.
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	1:7-11 (“The present invention relates to a system for distributing a software program through a network, and more specifically to a device and method for managing (controlling or administering) an identifier indicating information about a destination to which a software program is distributed.”) 1:61-65 (“The object of the present invention is to provide a device and a method for managing identifiers indicating a terminal and a user to which the software program is distributed, and for indicating the distributed software, used in the on-line software distribution system.”)
1-A	selecting a program residing in the volatile memory,	1:54-56 (“If the software program installed in the machine becomes unserviceable for some reason, a mechanism for restoring the software program is necessary for the on-line system”) 2:49-59 (“The identifier managing method according to the invention is used for managing a software program distributed to a terminal from a distribution center through a network. The identifier managing method includes the steps of, storing user information including a user identifier indicating a user to which the software program is distributed, storing terminal information including a terminal identifier indicating a terminal in which the software program is installed, and managing the user information and the terminal information by relating the user information with the terminal information.”) 4:59-5:3 (“The managing unit 1 manages the user identifier and the terminal identifier, relating the user identifier stored in the user information storage unit 4 with the terminal identifier stored in the terminal information storage unit 5. Therefore, it is possible to recognize which user receives the software program and which terminal the software program is installed in.

Exhibit 34**Claim Chart for U.S. Patent No. 6,411,941 based on Osborn (6,026,293)**

Element	Claim Language	Osborn
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	Abstract (“Methods and apparatus for preventing tampering with memory in an electronic device, such as a cellular telephone, are disclosed. An electronic device having a memory and a processing means contains logic that is used to perform a one-way hash calculation on the device’s memory contents whereby an audit hash value, or signature, of such contents is derived. The audit hash value is compared to an authenticated valid hash value derived from authentic memory contents. A difference between the audit and valid hash values can be indicative of memory tampering.”)
1-A	selecting a program residing in the volatile memory,	7:60-8:18 (“Referring to FIG. 4, a controller 400 controls operation of a cellular telephone (see, e.g., reference 1204 in FIG. 12). The controller 400 operates in conjunction with a flash program memory 420, an electronically erasable programmable read only memory (EEPROM) 410, and a random access memory (RAM) 408. The controller 400 includes a microprocessor 402 and an internal read-only memory (IROM) 403. The IROM 403 contains boot code 404, hashing code 405, authorization code 409, and a public encryption key 406. The controller also contains a protected static random access memory (PSRAM) 407, an interrupt controller 421, and hardware-based timers 401 for initiating periodic hash calculations by the microprocessor 402 on selected memory contents. The EEPROM 410 includes user profile data 412, an ESN 414, a MIN 416, and a signed/unsigned valid hash value pair 418. Instruction code involved with the general operation of the cellular telephone is contained in the flash program memory 420. The RAM Memory 408 is used as a scratchpad for operations which are part of the normal cellular telephone call processing. Operations involving sensitive data, hash value calculations and authentication processes are preferably carried out in conjunction with the PSRAM 407. The controller 400 communicates with the flash program memory 420, the RAM 408, and the EEPROM 410 via memory bus 424.”)

Exhibit 35

Claim Chart for U.S. Patent No. 6,411,941 based on Miller (U.S. Patent No. 6,038,320)

Element	Claim Language	Miller
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	1:4-8 (“The present invention relates to computer security and, more particularly, to a computer security key that is required to be coupled to a bus in a computer to enable access to that computer.”) 1:60-63 (“The computer 20, like the computer 10 in FIG. 1, includes a CPU 12, a BIOS flash memory 24 and a keyboard controller 18 coupled to a data/address bus 16.”)
1-A	selecting a program residing in the volatile memory,	2:59-3:7 (“FIG. 3 is a block diagram of an embodiment of the computer security key. The computer security key 40 includes a USB plug 48 that plugs into a single USB port, such as the USB port 32 or the USB port 31 in the computer 20 shown in FIG. 2. The computer security key 40 further includes a controller 42 coupled to the plug 48. The controller 42 is preferably an 82930A USB microcontroller from Intel Corporation, but it can be any type of controller that is capable of interfacing with the USB 28 and being programmed to execute the steps described herein. A key flash memory 46 is coupled to the controller 42. Each computer security key 40 is assigned a unique key code. The key code is stored in flash memory 46. Similarly, each computer 20 is assigned a key code that corresponds to a security key. The computer 20 stores the key code in its BIOS flash memory 24.”) 4:3-6 (“The program for performing the steps of FIG. 5, and all programs that are used by the computer 20 in conjunction with the key 40 can be stored on any computer-readable media.”) 5:10-11 (“After the computer 20 has completed its boot, the operating system is loaded and controls the computer 20.”)
1-B	using an agent to set up a	1:14-23 (“An example of computer security that is currently available is shown in

Exhibit 36**Claim Chart for U.S. Patent No. 6,411,941 based on Mirov (U.S. Patent No. 6,138,236)**

Element	Claim Language	Mirov
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	2:7-20 (“The present invention provides an apparatus for firmware authentication and methods for operating the same which result in software upgradability to firmware without compromising the integrity of the firmware. The novel application for authentication of firmware is based on cryptography. Thus, according to one aspect of the invention, a boot PROM (programmable read only memory) having programming instructions for initiating a computer system is provided. A software programmable data section has a plurality of micro-code. An authentication section having a hash generator generates a data hash in response to the plurality of micro-code programmed in the software programmable data section to authorize execution of the plurality of micro-code of the data section.”)
1-A	selecting a program residing in the volatile memory,	1:15-24 (“Micro-code, also referred to as firmware or boot code, is the first level of the instructional routines that are executed when the computer system is initially powered up. The micro-code stored in non-volatile memory devices such as a memory IC (integrated circuit) directs the computer system to certain boot blocks located on a disk drive. As these boot blocks on the disk drive are executed, successively larger blocks of boot data are loaded until finally the operating system, such as an Unix or Microsoft Windows of the computer system is loaded.”) 3:14-35 (“The invention will be described with respect to the Figures in which FIG. 1 generally shows a simplified computer system 10. The computer system 10 includes a CPU (central processing unit) 12, display 14, hard disk 16 and a flash PROM (programmable read-only memory) 18. The computer system 10 is for illustrative purposes as many variations to the architecture of the computer system 10 are available and known in the art. CPU bus 22 couples the CPU 12 to data bus 13. The CPU 12 includes a memory 15 which stores instructions and data for processing by the CPU 12. Disk drive bus 26 couples the disk drive 16 to the data bus 13. The disk drive 16 provides non-volatile data storage for the computer

Exhibit 37**Claim Chart for U.S. Patent No. 6,411,941 based on Fieres (6,148,083)**

Element	Claim Language	Fieres
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	7:49-56 (“Storage Elements. The cryptographic unit needs several storage elements. The flash memory 48 is the storage for programs and nonvolatile data stored in the cryptographic unit. The ROM storage 49 hosts the bootstrap code which executes on reset of the cryptographic unit. The RAM storage 50 holds the volatile data of the cryptographic unit.”) 7:32-37 (“Secure Execution. The cryptographic unit allows for the execution of code in the secure unit and tamper-resistant environment of that unit. Applications and subsystem layers may take advantage of this facility to implement a portion of their functionality, such as security protocol handlers, in this secure environment.”). Figs. 1, 4.
1-A	selecting a program residing in the volatile memory,	10:54-60 (“For critical applications, there has long been a need to validate that an application has not been tampered with. Performing this validation usually involves a trusted load subsystem. A trusted load subsystem is the part of the operating system which is responsible for loading a program image into the system memory space and while doing that validate that the application has not been tampered with.”) 12:28-30 (“In the execution stage, an application is loaded by the operating system into the memory system and starts execution.”) 13:4-10 (“The scheme described above is readily extended to cover not only code images, but also any kind of data object which can make use of the validation method outlined. For example, the operating system itself, subsystem libraries, and

Exhibit 38**Claim Chart for U.S. Patent No. 6,411,941 based on Schwartz (U.S. Patent No. 6,153,835)**

Element	Claim Language	Schwartz
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	Abstract (“In an improved postage scale system, soft-selection keys are used for selecting options including shipping service options provided by the system. The display on such a system is adjustable for comfortable viewing thereof. The system is capable of interfacing and communicating with other devices such as a printer, a remote computer, an optical scanner, an integrated circuit (IC) card, etc. Certain data and code for use in the system can be updated by downloading new data and code from IC cards to the system. A security measure is instituted such that, after an update of the data or code, a system user is required to enter a valid authorization number in order to enable the system. Such a measure is, in part, necessary for deterring unauthorized copying of software of the system onto another similar system.”) 7:48-59 (“In addition, console 13 includes read-only-memory (ROM) 213 permanently storing a unique serial number pre-assigned to system 10. Console 13 also includes memory section 250 comprising flash electrically erasable programmable read-only-memory (EEPROM) 250a having a capacity of 128 Kbytes, EPROM 250b having a capacity of 32 Kbytes, a nonvolatile static random-access-memory (SRAM) 250c having a capacity of 128 Kbytes, and SRAM 250d having a capacity of 128 Kbytes, all of which are of conventional design. Flash EEPROM 250a may illustratively be an AT29C010 memory manufactured and marketed by Atmel Corporation. For technical details on the specification and functionality of the AT29C010, one may refer to "ATMEL CMOS Data Book," Atmel Corporation, 1991-1992.”) 8:13-15 (“Basic Input/Output System (BIOS) module 309 contains firmware responsible for basic machine operation of system 10.”)

Exhibit 39

Claim Chart for U.S. Patent No. 6,411,941 based on Arbaugh (U.S. Patent No. 6,185,678)

Element	Claim Language	Arbaugh
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	4:33-59 (“The present invention can also be utilized to reduce the Total Cost of Ownership (TCO) of a personal computer, through automatically detecting and repairing integrity failures, thereby permitting the user to continue to work without the nuisance of a trouble call to support staff and the associated down time. A log can be created by the trusted repository of the present invention which can be monitored by a system administrator to identify workstations that require "hands on" repairs, e.g. ROM failure, enabling the system administrator to schedule the work to be done when the user is not using the computer. The present invention also enables the bootstrap components to be automatically updated. One way this can be done is to limit the validity period of the cryptographic certificates associated with each component of the bootstrap process. When the certificate expires, the trusted repository of the present invention is contacted and either a new certificate is obtained, in the case where the component does not need an update, or a new component and certificate are obtained, in the case where a newer version of the component is available. This permits the system administrator to update all of the workstations from a central location without having to visit each individual computer system. A second approach is to add a hook to the BIOS to contact the trusted repository at the beginning of the bootstrap process of the present invention. The purpose of this contact is two fold. First, it permits a status monitoring of each workstation. Second, the contact allows centralized updates to be done in the following manner. When the server receives the "I am booting" message from the client, the server would check a database containing the configuration of the client. The Server would then compare that configuration with the current configuration desired for the client. If they are different, then the server would instruct the client to download the appropriate changes.”) 6:7-25 (“The present invention is an architecture for initializing a computer system.

Exhibit 40**Claim Chart for U.S. Patent No. 6,411,941 based on Misra et al. (U.S. Patent No. 6,189,146)**

Element	Claim Language	Misra et al.
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	Abstract (“A software licensing system includes a license generator located at a licensing clearinghouse and at least one license server and multiple clients located at a company or entity. When a company wants a software license, it sends a purchase request (and appropriate fee) to the licensing clearinghouse. The license generator at the clearinghouse creates a license pack containing a set of one or more individual software licenses. To prevent the license pack from being copied and installed on multiple license servers, the license generator assigns a unique license pack ID to the license pack and associates the license pack ID with the particular license server in a master license database kept at the licensing clearinghouse. The license generator digitally signs the license pack and encrypts it with the license server's public key. The license server is responsible for distributing the software licenses from the license pack to individual clients. When a client needs a license, the license server determines the client's operating system platform and grants the appropriate license. To prevent an issued license from being copied from one client machine to another, the software license is assigned to a specific client by including a client ID within the license. The software license also has a license ID that is associated with the client ID in a database record kept at the license server. The license server digitally signs the software license and encrypts it using the client's public key. The license is stored locally at the client.”) 1:6-8 (“This invention relates to systems and methods for licensing software. This invention further relates to systems and methods for enforcing software licenses.”) 2:12-16 (“This invention concerns a system and method for licensing software. The system and method provides confidence to the vendor that the software license is being complied with, while also assisting the purchaser in monitoring its own compliance with the license.”)

sf-3054996

1

Exhibit 41**Claim Chart for U.S. Patent No. 6,411,941 based on Saunders (6,209,099)**

Element	Claim Language	Saunders
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	Abstract (“The present invention concerns a secure data processing method and system in which the user or operator of the system can trust that all of the software and hardware components of the system have been authenticated The cryptographic engine operates on the contents of the cryptographic key store to generate a digital signature. Means are provided to generate a digital signature from a software or hardware component to be checked for authenticity and to compare the digital signature from the component with the generated digital signature. An indication of the authenticity of the component is generated as a result of the comparison. The components of the system that can be checked include the boot firmware for the system, the operating system and plug-in cards for the system.”)
1-A	selecting a program residing in the volatile memory,	2:43-46 (“The components to be authenticated in the system of FIG. 1 include the operating system (OS), the firmware on the plug-in cards 13, and the boot firmware 16. The invention may be applied to a system which has either more or less system components to be authenticated than the system depicted in FIG. 1.”) 3:62-64 (“If the boot record is valid, the boot record code is executed in step 40 and the system is running.”) Figs. 1, 2, 3.
1-B	using an agent to set up a verification structure in the erasable, non-volatile memory of the BIOS, the verification structure accommodating data that includes at least one license	2:52-60 (“Each of the components of the system which are to be authenticated includes a digital signature which is embedded in the firmware of the component. The digital signature is embedded at a predefined location and is created by the supplier of the component as part of the manufacturing process. The algorithm for generating the digital signatures uses an symmetric key pair, with the vendor supplier keeping the private key secure and distributing the public key with the

Exhibit 42**Claim Chart for U.S. Patent No. 6,411,941 based on Pearce et al. (U.S. Patent No. 6,243,468)**

Element	Claim Language	Pearce
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	1:6-12 (“This invention relates to systems and methods for preventing piracy or illicit use of software. More particularly, this invention relates to such systems and methods that allow hardware components of the underlying computer to be upgraded and the software to be legitimately installed on the upgraded machine without triggering the anti-piracy protection.”) 2:35-48 (“This invention concerns an anti-piracy system and method that reduces the opportunity for piracy and illicit use of software products by requiring each software product to be registered for a single computer that consists of a specific hardware composition. If a user attempts to install the software product on another computer, the software product will recognize a different hardware composition and disable itself According to one aspect of the invention, the system includes a software product that is loaded onto a specific computer having a set of hardware components (e.g., RAM, hard disk drive, floppy disk drive, BIOS, network card, video card, etc.).”)
1-A	selecting a program residing in the volatile memory,	3:5-7 (“Each time the software product is subsequently launched, the software product again obtains the product ID and generates the hardware ID for the computer.”) 4:27-67 (“FIG. 3 shows the customer computer 32 within an exemplary implementation of a personal computer. The computer 32 includes the processing unit 40, system memory 42, and a system bus 44 that interconnects various system components, including the system memory 42 to the processing unit 40. The system bus 44 may be implemented as any one of several bus structures and using any of a variety of bus architectures, including a memory bus or memory controller, a peripheral bus, and a local bus. The system memory 42 includes read only

sf-3054982

1

Exhibit 43**Claim Chart for U.S. Patent No. 6,411,941 based on Cotichini (U.S. Patent No. 6,269,392)**

Element	Claim Language	Cotichini
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	Abstract (“A system for locating and monitoring electronic devices utilizing a security system that is secretly and transparently embedded within the software, firmware, or hardware of the electronic device. The security system may be embodied in internal memory devices such as ROM BIOS, ROM, Flash ROM, EPROM, EEPROM, electronic components such as modem, CPU, or hardwired circuits or integrated circuits of a computer. This security system initiates the computer to periodically and conditionally call a host system to provide unique identifying indicia and location information. In one embodiment, the security system calls the host through a public switched telephone network (PSTN) and transmits the indicia in encoded form. In an alternative embodiment, which may be incorporated concurrently with the PSTN application, the security system calls the host system through the Internet and provides the host with indicia encoded within the DNS query sent. The host system is able to identify the calling computer from the indicia and is able to physically locate the computer through either caller identification or by referencing the Internet links which were used to connect the calling computer with the host system.”)
1-A	selecting a program residing in the volatile memory,	11:62-12:17 (“Referring to FIG. 5, the Agent startup and loading sequence is described. The computer 10 is powered on and the loading sequence begins 700. As is well known in the art, the machine's start-up ROM procedure 701 in the computer 10 begins when the power is turned on. This process supervises the booting up and loading of the operating system of the computer. It performs the power-on self-test (POST) 702, in fact POST is carried out on every reset of the system, including the time when the power is first turned on. This test has two purposes: it performs a quick test 703 of the basic elements of the system; and it initializes the major hardware components for use. POST tests all of the ROMs on

Exhibit 44**Claim Chart for U.S. Patent No. 6,411,941 based on Pavlin (U.S. Patent No. 6,523,119)**

Element	Claim Language	Pavlin
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	4:49-5:24 (“FIG. 2 presents a block diagram describing the overall architecture of the present invention. The host computer 100 is coupled to a hardware key 114 providing a host computer/hardware key communications link 138. This communications link may be any computer I/O link, including a parallel printer port, an RS-232 or RS-423 serial port, a Macintosh.TM. ADB port, or any port used for remote or network communications. The host computer 100 responds to the instructions in one or more application programs 120 to process data and provide output results. The present invention prevents unauthorized use or copying of application programs 120 in the host computer 100, or any other processor coupled via the external communications link 110. Application programs 120 communicate with the hardware key 114 via an application program interface 121 (API), client library 123, and an interface driver 122. The hardware key 114 includes a custom designed application-specific integrated circuit (ASIC) 128. The ASIC 128 is coupled to the host computer 100 via the hardware key interface port 126 and host computer/hardware key communications link 138. ASIC 128 contains a microprocessor 130, and associated microprocessor RAM 131, a firmware read only memory (F/W ROM) 136, random access memory (RAM) 132 and a developer-configurable electronically erasable programmable read only memory (EEPROM) 134. Of course, EEPROM 134 can be replaced by any programmable non-volatile memory technology, including anti-fuse read only memory technologies. These anti-fuse technologies offer one-time write capability and do not allow external inspection of data bit states. Co-locating these elements in a single ASIC 128 allows critical security-related values to be stored in memories such as the EEPROM 134 which are difficult to penetrate from outside

Exhibit 45

Claim Chart for U.S. Patent No. 6,411,941 based on Hannah (U.S. Patent No. 6,735,696)

Element	Claim Language	Hannah
Claim 1		
1-P	1. A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of:	Abstract (“A method and system of protecting content by ensuring a secure boot of a processor environment. A processor is coupled to a certification unit which may include a decryption engine and a one-way hash unit. The certification unit is employed to ensure that only certified software is permitted access to memory containing content and that only certified software may access a content bit stream such as may flow across a 1394 serial bus from a digital video disk player. By preventing access by uncertified software, the risk of a personal computer being used as an instrumentality of piracy is reduced.”) 1:65-2:25 (“FIG. 1 is a block diagram of the system of one embodiment of the invention. A processor core 10 is coupled to a memory 22 by a bus 32. Memory 22 is partitioned into operating system (O/S) protected memory 24 which is deemed uncopiable by applications. In the remainder of the memory, which may be freely copied and/or accessed by applications executing on the processor core, a software certification unit 28 is provided to ensure that software executing on the processor core 10 is certified. Certification unit 28 includes an decryption engine 12, a hash unit 14, and a comparator 16. A flash key storage unit 18 is coupled to the decryption engine 12 to provide it with keys necessary for it to perform its decryption function. In one embodiment, decryption engine 12 will perform public key decryption and verification of digital signatures. A flash unit 20 provides authorized hash values to one input of comparator 16 as part of the certification process. Flash unit 18 and flash unit 20 may be a single flash memory unit. A digital video disk player (DVD) 26 is coupled to memory by a serial bus 34 which in one embodiment of the invention implements a 1394 protocol. In operation, a content bit stream either encrypted or unencrypted will flow across the serial bus 34 to memory 22. In one embodiment, the bit stream will be stored in protected memory 24. In that manner, if the O/S is not compromised, the content cannot be

EXHIBIT 46**ADDITIONAL PRIOR ART RELEVANT TO THE INVALIDITY OF THE '941 PATENT**

Author	Non-U.S. Patent Publication	Publication Date
Purdy et al.	A Software Protection Scheme, Proc. 1982 Symp. On Security and Privacy, pp. 99-103	1982
Maude et al.	Hardware Protection Against Software Piracy, Communications of the ACM, pp. 950-959	1984
Albert et al.	Combatting [sic] Software Piracy by Encryption and Key Management, IEEE Computer, pp. 68-73	April 1984
Hoff et al.	System and Software via Authentication Handshake in EPROM, Proceedings of the National Computer Conference, pp. 203-209	1985
Herzberg et al.	Public Protection of Software, ACM Transactions on Computer Systems, pp. 371-393	Nov. 1987
Davida	Defending Systems Against Viruses through Cryptographic Authentication, Proceedings IEEE Symposium on Security and Privacy, pp. 312-318	1989
Gasser et al.	The Digital Distributed System Security Architecture, Proceedings of 1989 National Computer Security Conference	1989
Tygar et al.	Strongbox: A System for Self-Securing Programs, Carnegie Mellon Computer Science: A 25-Year Commemorative, pp. 163-97	1991
Lampson et al.	Authentication in Distributed Systems: Theory and Practice, ACM Trans. Computer Systems, pp. 265-310	Nov. 1992
Spesivtsev et al.	Software Copy Protection Systems: Structure, Analysis, Attack, Proceedings IEEE 1992 International Carnahan Conference on Security Technology: Crime Countermeasures, pp. 179-182	Oct. 1992
Hartig et al.	The BirliX Security Architecture, Journal of Computer Security , pp. 3-16	1993

sf-3058449

Hauser et al.	LTTP Protection – A Pragmatic Approach to Licensing, Institut fur Informatik Universitat Zurich	Jan. 1994
Kim et al.	The Design and Implementation of Tripwire: A File System Integrity Checker	Feb. 23, 1995
Gobioff et al.	Smart Cards in Hostile Environments, Carnegie Mellon University Computer Science Technical Report CMU-CS-95-188	Sept. 1995
Dornbusch et al.	Desktop Management Software; Total Control No Need to Adjust Your Set, InfoWorld	Sept. 1995
Emerson	Fighting Software Piracy: What Are the Alternatives, Computing Japan	May 1996
Kuhn	The Trust No 1 Cryptoprocessor Concept	Apr. 30, 1997
Arbaugh	Automated Recovery in a Secure Bootstrap Process	Nov. 21, 1997
Phoenix Techs. Ltd./RSA Data Security	RSA and Phoenix Technologies Combine to Bring Preboot Security to Personal Computers; Partnership Aims to Plug Long Neglected PC System Security Gap, Business Wire	Jan. 12, 1998
Gilmont et al.	An Architecture of Security Management Unit for Safe Housing of Multiple Agents In Proceedings of the International Workshop of Intelligent Communications and Multimedia terminals	1998
Alexander et al.	Safety and Security of Programmable Network Infrastructures, Communications Magazine IEEE, pp. 84-92	Oct. 1998
Tygar et al.	Cryptographic Postage Indicia, CMU-CS-96-113	January 1996
Inventor	Patent Number	Issue Date
Chorley et al.	U.S. 4,634,807	Jan. 6, 1987
Chandra et al.	U.S. 4,644,493	Feb. 17, 1987
Dunham et al.	U.S. 4,791,565	Dec. 13, 1988

sf-3058449

Wolfe	U.S. 4,796,220	Jan. 3, 1989
Earnest	U.S. 4,888,798	Dec. 19, 1989
Chandra et al.	U.S. 4,903,296	Feb. 20, 1990
Hershey et al.	U.S. 4,924,378	May 8, 1990
Cina	U.S. 4,930,073	May 29, 1990
Robert et al.	U.S. 4,937,863	Jun. 26, 1990
Dyson	U.S. 5,050,212	Sep. 17, 1991
Chou et al.	U.S. 5,081,676	Jan. 14, 1992
Waite et al.	U.S. 5,103,476	Apr. 7, 1992
Comerford et al.	U.S. 5,109,413	Apr. 28, 1992
Durst, Jr. et al.	U.S. 5,113,518	May 12, 1992
Mizuta	U.S. 5,131,091	Jul. 14, 1992
Corbin	U.S. 5,138,712	Aug. 11, 1992
Medveczky et al.	U.S. 5,182,770	Jan. 26, 1993
Logan	U.S. 5,199,066	Mar. 30, 1993
Graves et al.	U.S. 5,309,363	May 3, 1994
Smyth et al.	U.S. 5,325,430	Jun. 28, 1994
Chou et al.	U.S. 5,337,357	Aug. 9, 1994
Lassers	U.S. 5,343,526	Aug. 30, 1994
Arnold	U.S. 5,379,342	Jan. 3, 1995
Christiano	U.S. 5,386,369	Jan. 31, 1995
Barber et al.	U.S. 5,390,297	Feb. 14, 1995
Shear	U.S. 5,410,598	Apr. 25, 1995
Cane et al.	U.S. 5,416,840	May 16, 1995

sf-3058449

Ancora's Exhibit 2004
HTC v. Ancora IPR2021-00570
Page 70 of 74

Jablon et al.	U.S. 5,421,006	May 30, 1995
Wyman	U.S. 5,438,508	Aug. 1, 1995
Clark	U.S. 5,448,045	Sep. 5, 1995
Bergum et al.	U.S. 5,457,748	Oct. 10, 1995
Davis	U.S. 5,473,692	Dec. 5, 1995
Watson et al.	U.S. 5,475,839	Dec. 12, 1995
Ewertz et al.	U.S. 5,479,639	Dec. 26, 1995
Benson et al.	U.S. 5,541,991	Jul. 30, 1996
Findley et al.	U.S. 5,629,508	May 13, 1997
Christiano	U.S. 5,671,412	Sep. 23, 1997
Schwartz et al.	U.S. 5,675,645	Oct. 7, 1997
Goldman et al.	U.S. 5,684,951	Nov. 4, 1997
Rose	U.S. 5,708,709	Jan. 13, 1998
Rosenow et al.	U.S. 5,724,426	Mar. 3, 1998
Dunham et al.	U.S. 5,742,758	Apr. 21, 1998
Angelo et al.	U.S. 5,748,940	May 5, 1998
Bereiter	U.S. 5,754,763	May 19, 1998
Brandt et al.	U.S. 5,758,068	May 26, 1998
Olsen	U.S. 5,758,069	May 26, 1998
Cain et al.	U.S. 5,764,892	Jun. 9, 1998
Mattison	U.S. 5,778,070	Jul. 7, 1998
Coley et al.	U.S. 5,790,664	Aug. 4, 1998
Nagashima	U.S. 5,796,940	Aug. 18, 1998
Cotichini et al.	U.S. 5,802,280	Sep. 1, 1998

sf-3058449

Ancora's Exhibit 2004
HTC v. Ancora IPR2021-00570
Page 71 of 74

Archibald et al.	U.S. 5,825,883	Oct. 20, 1998
Chou et al.	U.S. 5,826,011	Oct. 20, 1998
Cruts et al.	U.S. 4,780,905	Oct. 25, 1998
Albrecht et al.	U.S. 5,835,594	Nov. 10, 1998
Davis	U.S. 5,844,986	Dec. 1, 1998
Ginter et al.	U.S. 5,892,900	Apr. 6, 1999
Chou et al.	U.S. 5,892,906	Apr. 6, 1999
Angelo	U.S. 5,944,821	Aug. 31, 1999
Olsen et al.	U.S. 5,905,860	May 18, 1999
Ahmad	U.S. 5,925,127	Jul. 20, 1999
Ginter et al.	U.S. 5,917,912	Jun. 29, 1999
Morisawa et al.	U.S. 5,931,948	Aug. 3, 1999
Scheck et al.	U.S. 5,933,498	Aug. 3, 1999
Hasebe et al.	U.S. 5,935,243	Aug. 10, 1999
Griswold	U.S. 5,940,504	Aug. 17, 1999
Shiell et al.	U.S. 5,950,012	Sep. 7, 1999
Scott et al.	U.S. 5,987,123	Nov. 16, 1999
Shin et al.	U.S. 5,987,134	Nov. 16, 1999
Steinberg et al.	U.S. 6,000,030	Dec. 7, 1999
Baena-Arnaiz et al.	U.S. 6,006,190	Dec. 21, 1999
Duvvoori et al.	U.S. 6,021,438	Feb. 1, 2000
Grumstrup et al.	U.S. 6,023,763	Feb. 8, 2000
Osborn	U.S. 6,026,293	Feb. 15, 2000
Bhatia et al.	U.S. 6,052,803	Apr. 18, 2000

sf-3058449

Ancora's Exhibit 2004
HTC v. Ancora IPR2021-00570
Page 72 of 74

Fette et al.	U.S. 6,052,600	Apr. 18, 2000
Glover	U.S. 6,052,780	Apr. 18, 2000
Horstmann	U.S. 6,055,503	Apr. 25, 2000
Baltz et al.	U.S. 6,058,474	May 2, 2000
Smith et al.	U.S. 6,067,582	May 23, 2000
Sesma	U.S. 6,073,256	Jun. 6, 2000
Knutson	U.S. 6,078,909	Jun. 20, 2000
Christenson et al.	U.S. 6,122,733	Sep. 19, 2000
Goetz et al.	U.S. 6,128,741	Oct. 3, 2000
Fieres et al.	U.S. 6,148,083	Nov. 14, 2000
Khan et al.	U.S. 6,173,446	Jan. 9, 2001
Nardone et al.	U.S. 6,175,925	Jan. 16, 2001
Garst et al.	U.S. 6,188,995	Feb. 13, 2001
Barr et al.	U.S. 6,189,100	Feb. 13, 2001
Wallace	U.S. 6,192,475	Feb. 20, 2001
Edenson et al.	U.S. 6,198,875	Mar. 6, 2001
Saunders	U.S. 6,209,099	Mar. 27, 2001
Reardon	U.S. 6,212,635	Apr. 3, 2001
Larsson et al.	U.S. 6,226,747	May 1, 2001
Cohen	U.S. 6,233,567	May 15, 2001
Solymar et al.	U.S. 6,244,758	Jun. 12, 2001
Galasso et al.	U.S. 2001/0007131 A1	Jul. 5, 2001
Narayanaswamy et al.	U.S. 6,275,931	Aug. 14, 2001

sf-3058449

Ancora's Exhibit 2004
HTC v. Ancora IPR2021-00570
Page 73 of 74

Gotoh et al.	U.S. 6,298,138	Oct. 2001
Cotichini et al.	U.S. 6,300,863	Oct. 9, 2001
Neville et al.	U.S. 6,272,636	Aug. 7, 2001
Eyres et al.	U.S. 6,324,649	Nov. 27, 2001
Davis et al.	U.S. 6,401,208	Jun. 4, 2002
Cain et al.	U.S. 6,507,914	Jan. 14, 2003
Kawamura et al.	U.S. 6,519,701	Feb. 11, 2003
Shimizu et al.	U.S. 6,684,198	Jan. 27, 2004
Ishiguro et al.	U.S. 6,697,945	Feb. 24, 2004
Saito	U.S. 6,789,197	Sep. 7, 2004
Pavlin et al.	U.S. 7,024,564	Apr. 4, 2006
Alcorn et al.	U.S. RE 39,400E	Nov. 14, 2006
Barber et al.	WO 89/04520	May 18, 1989
Richardson	WO 94/07204	Mar. 31, 1994
Kunihiro et al.	JP 08286906 A	Jan. 11, 1996

sf-3058449

Ancora's Exhibit 2004
HTC v. Ancora IPR2021-00570
Page 74 of 74