UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

KINGSTON TECHNOLOGY COMPANY, INC.,
Petitioner,

v.

MEMORY TECHNOLOGIES, LLC,
Patent Owner

Case No.: To Be Assigned
U.S. Patent No. RE45,486

PETITION FOR *INTER PARTES* REVIEW OF U.S. PATENT NO. RE45,486

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

TABLE OF CONTENTS

TABLE OF AUTHORITIES

Page(s)

Cases

Statutes and Codes

Rules and Regulations

EXHIBIT LIST

Exhibit No.	***Description***
1001	United States Patent No. RE45,486 (the "RE486 Patent")
1002	File History for U.S. Patent No. RE45,486
1003	United States Patent No. 7,257,669 (the "'669 Patent")
1004	File History for United States Patent No. 7,257,669
1005	United States Patent No. 6,279,114 to Toombs et al.
1006	United States Patent No. 6,314,504 to Dent
1007	PC Card Standard, Volume 2 Electrical Specification
1008	The AT Attachment with Packet Interface - 6 Standard, Revision 3a
1009	Declaration of Dr. R. Jacob Baker
1010	The MultiMediaCard System Specification, Version 1.4 ("MMC Specification")
1011	United States Patent No. 6,260,101 to Hanzen
1012	Structured Computer Organization, Fourth Edition, by Andrew S. Tannenbaum
1013	United States Patent No. 6,253,300 to Lawrence et al.
1014	United States Patent Publication No. 2003/0235408 to Silvester et al.
1015	Affidavit of Christopher Butler, Internet Archive

I. INTRODUCTION AND STATEMENT OF RELIEF REQUESTED (37 C.F.R. §42.22(a))

Kingston Technology Company, Inc. ("Petitioner" or "Kingston") hereby petitions to institute an *inter partes* review of Claims 6, 8-11, 23, 25-27, and 30-31 (the "Challenged Claims") of U.S. Patent No. RE45,486 (the "RE486 Patent") to Ahvenainen *et al*. (Ex. 1001), and cancel those claims as unpatentable.

As discussed below, the prior art anticipates and/or renders obvious Claims 6, 8-11, 23, 25-27, and 30-31 of the RE486 Patent under 35 U.S.C. § 102 and/or § 103. Accordingly, there is a reasonable likelihood that Petitioner will prevail with respect to at least one challenged claim, and Petitioner respectfully requests that the Board institute a trial for *inter partes* review and cancel all Challenged Claims as unpatentable.

II. MANDATORY NOTICES (37 C.F.R. § 42.8(a)(1)

A. Real Party-In-Interest (37 C.F.R. § 42.8(b)(1))

Petitioner Kingston Technology Company, Inc., is a real party-in-interest. Petitioner's parent company, Kingston Technology Corporation ("Kingston Holding"), is a holding company without any employees or operations. However, because Kingston Holding is the sole owner of Petitioner and shares some directors, Petitioner identifies Kingston Holding as an additional real party-in-interest.

B. Identification of Related Matters (37 C.F.R. § 42.8(b)(2))

Patent Owner Memory Technologies, LLC ("MTL") has asserted the Challenged Claims of the RE486 Patent, as well as claims from seven other patents, against Kingston in a co-pending litigation, *Memory Technologies, LLC v. Kingston Technology Co., Inc*., 8:18-cv-00171 (C.D. Cal.). MTL's original Complaint was filed on January 31, 2018, and served, at the earliest, on February 1, 2018.

In addition to this Petition, Kingston has or will be filing petitions for *inter partes* review of the other seven patents that MTL has asserted against it.

C. Lead and Back-Up Counsel Under 37 C.F.R. § 42.8(b)(3)

Petitioner designates the following Lead and Backup Counsel. Concurrently filed with this Petition is a Power of Attorney for appointing the following Lead and Backup Counsel, per 37 C.F.R. § 42.10(b). Service via hand-delivery may be made at the postal mailing addresses below. Petitioner consents to electronic service by e-mail at the following address: **kingston-RE486ipr@pillsburylaw.com.**

Lead Counsel	Back-Up Counsel
Robert C.F. Pérez (Reg. No. 39,328) PILLSBURY WINTHROP SHAW PITTMAN LLP 1650 Tysons Boulevard, 14th Floor McLean, VA 22101 Telephone: 703.770.7900 Facsimile: 703.770.7901	**Christopher Kao**, Kingston's counsel in the co-pending litigation (*Pro hac vice* motion to be filed) **Brock S. Weber**, Kingston's counsel in the co-pending litigation (*Pro hac vice* motion to be filed) PILLSBURY WINTHROP SHAW PITTMAN LLP Four Embarcadero Center, 22nd Floor San Francisco, CA 94111 Telephone: 415.983.1000 Facsimile: 415.983.1200

D. Payment of fees (37 C.F.R. §42.103)

Petitioner authorizes the Patent and Trademark Office to charge Deposit Account No. 033975 for the petition fee and for any other required fees.

III. REQUIREMENTS FOR *INTER PARTES* REVIEW

A. Identification of Challenge (37 C.F.R. § 42.104(b))

Pursuant to 37 C.F.R. §42.104(b), Petitioner requests that Claims 6, 8-11, 23, 25-27, and 30-31 of the RE486 Patent be cancelled as anticipated or rendered obvious based on the following grounds:

Ground	RE486 Patent Claims	Basis for Rejection
1	6, 8-11, 25-27, and 30-31	§ 102 based on Toombs
2	6, 8-11, 23, 25-27, and 30-31	§ 103 based on Toombs and Dent
3	6, 8-11, 23, 25-27, and 30-31	§ 103 based on Toombs and PCMCIA

4	6, 8-11, 23, 25-27, and 30-31	§ 103 based on Toombs and Revision 3a of ATA-6

The Declaration of Jacob Baker, Ph.D., P.E, filed herewith (Ex. 1009, "Baker Decl."), supports the challenge in this Petition that the Challenged Claims are invalid as anticipated and obvious.

B. Patents and Publications Relied Upon

United States Patent No. 6,279,114 to Toombs et al. ("Toombs") was filed on November 4, 1998 and issued on August 21, 2001. Toombs incorporates by reference The MultiMediaCard System Specification, Version 1.4 by the MMCA Technical Committee ("MMC Specification"). Ex. 1005 at 31:18-22. Toombs is prior art under at least pre-AIA 35 U.S.C. §§ 102(a), (b), and (e). Toombs was not considered by the PTO during prosecution of the RE486 Patent.

United States Patent No. Dent 6,314,504 to Dent ("Dent") was on filed March 9, 1999 and issued and published on November 6, 2001. Dent is prior art under at least pre-AIA 35 U.S.C. §§ 102(a), (b), and (e). Dent was not considered by the PTO during prosecution of the RE486 Patent.

PC Card Standard, Volume 2 Electrical Specification ("the PCMCIA standard") is a publication that bears a copyright date of 1999 and was publicly available online from the Personal Computer Memory Card International Association (PCMCIA) website as of at least November 2001. Ex. 1007, at 2;

Ex. 1015 at 1-2, 17-26. Public availability of the PCMCIA standard is proven using the Internet Archive, described in detail below. The PCMCIA standard is prior art under at least pre-AIA 35 U.S.C. §§ 102(a) and (b). The PCMCIA standard was not considered by the PTO during prosecution of the RE486 Patent.

The AT Attachment with Packet Interface - 6 Standard, Revision 3a ("Revision 3a of the ATA-6 Standard") is a publication that was first published at least as early as January 2002. Revision 3a of the ATA-6 Standard was publicly available online from the T13 website as of at least January 2002. Ex. 1008; Ex. 1015 at 1-2, 16. Public availability of the PCMCIA standard is proven using the Internet Archive, described in detail below. Revision 3a of the ATA-6 Standard is prior art under at least pre-AIA 35 U.S.C. §§ 102(a) and (b). Revision 3a of the ATA-6 Standard was not considered by the PTO during prosecution of the RE486 Patent.

IV. BACKGROUND OF THE TECHNOLOGY

Memory cards, such as PC cards, compact flash ("CF") cards, secure digital ("SD") cards, or multimedia cards ("MMC"), are electronic data storage devices used in various portable electronic devices such as PDAs, cameras, and smart phones. Ex. 1005 at 1:19-22. Data is stored on a memory device by recording the data in memory cells, such as flash or EEPROM. *Id.* at 1:30-35.

Unveiled in 1997, the MMC standard governs certain solid-state storage memory cards based on a surface contact low pin-count serial interface. Ex. 1010. A MMC is controlled by commands, which are related to operations such as data read, data write, changing card status, or obtaining card information. Ex. 1005 at Figs. 38-44. Many commands use an address as a command argument to identify a location on the memory card to be accessed. Ex. 1005 at 9:51-55; Ex. 1009 at ¶80.

Addressing generally enables the access of data. Ex. 1009 at ¶81. Over time, as computer memory expanded, there arose a need to incorporate expanded methods of addressing into existing addressing methods. *Id.* Also common was the need to access addressing at different levels of granularity. *Id.* Of course, "in order to gain a finer memory resolution, one must pay the price of longer addresses and thus longer instructions." Ex. 1012 at 47; Ex. 1009 at ¶87. There were multiple approaches to solving this challenge.

Computer devices often organized memory in a hierarchy with relatively expensive high-speed devices, typically close to the processor, and relatively inexpensive lower speed but higher capacity speed, further from the processor. Ex. 1009 at ¶84. To manage these hierarchal arrangements of different data stores, computer systems have employed addressing methods including virtual addressing and paging, cache management, and a variety of different hardware and software

mechanisms to ensure the consistency of data stored sometimes in multiple locations within these memory hierarchies. Ex. 1012; Ex. 1009 at ¶¶83-89.

One solution was the use of higher and lower order bits for addressing. *Id.* at ¶85. A person of ordinary skill in the art ("POSITA") would know that addressing based on a subset of bits and adjusting the size of the chunk of data that is being accessed is a standard design technique. Ex. 1012 at 53-61. In the case of caches for example, a small, fast memory—the cache—is used to store heavily used memory words for access by the CPU in order to improve the speed at which the computer functions. *Id.* at 21. The cache is divided into fixed-size blocks, referred to as cache lines. *Id.* at 23. A cache manages memory addressing using a subset of address bits (cache tag) to determine whether a particular cache line is present in the cache or not. *Id.* The lowest order bits, or block index bits, can be used by a system to point to an individual piece of data within a cache line, but with regard to accesses to main memory, the cache ignores these bits because accesses are performed only at the level of granularity of an entire block or cache line. *Id.* at 23-24. For example, in the case of a 32-byte cache line size, the lowest order bits 0-4 are not part of the addressing method since the cache only accesses chunks of data that are 32 bytes long and aligned by a 32-byte block size. Ex. 1009 at ¶85.

Computer systems typically manage main system memory using the same concept. Ex. 1009 at ¶89. Memory is subdivided into chunks of consecutive addresses known as pages. Ex. 1012 at 51. Memory addresses are often broken into a virtual page number and a page offset. *Id.* at 52. Because the physical memory is often much smaller than the total address space, a computer will swap pages of data into and out of physical memory and maintain a table (known as the page table) by which the operating system can tell which pages of data are actually resident in the machine's memory. *Id.* at 51-52. In accessing data, the highest order bits of a virtual address are known as the virtual page number and the system translates virtual page numbers into physical pages. *Id.* at 52. The choice of the page size is an engineering trade-off that balances the efficiency of memory utilization (small page size) against the size of the data structure needed for addressing pages. Ex. 1009 at ¶89.

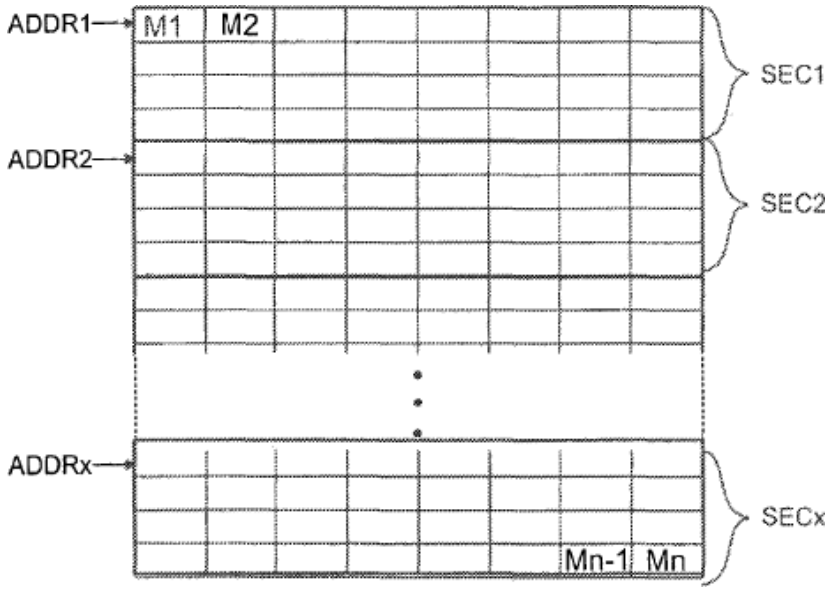
V. OVERVIEW OF THE RE486 Patent

A. Summary of the Claimed Subject Matter

The RE486 Patent notes that "in some memory card standards an upper limit has been set for the number of memory locations included in a memory card." Ex. 1001 at 2:19-21. But the amount of memory that can be fitted into a memory card was "larger than the upper limit determined by many standards." *Id.* at 2:21-26.

The RE486 Patent refers to the MMC specification in force at the time, stating: "in order to address the memory locations of a memory card according to the specifications of [a] Multimedia Card, there are 32 bits that can be used, with which a maximum of 4 gigabytes of memory space can be addressed." *Id.* at 2:31-34. The memory capacity of a memory card according to the MMC specification was calculated by multiplying the number of memory blocks with the length of the memory block. *Id.* at 2:42-44; Ex. 1009 at ¶95. The RE486 Patent asserts that the maximum capacity of a memory card according to the MMC specification was 4 gigabytes. Ex. 1001 at 2:62-67.

The RE486 Patent describes a method in which "two or more memory locations are addressed with one address, and/or the number of bits that can be used in an address is increased." *Id.* at Abstract. In one "sector-by-sector embodiment[,] one address ADDR1 addresses the data of one sector SEC1, with the next address ADDR2 to the data of the next sector SEC2, etc." *Id.* at 5:41-46. Thus, "with one … address the data of the memory locations M1, M2, ..., Mn of one sector is read." *Id.* at 5:46-48. Figure 2 of the RE486 Patent illustrates this embodiment:



The size of the sector may vary in different situations. *Id.* at 5:50-51. Further, "the size of the sector is not necessarily the same as the size of the [memory] block, but it can be smaller or larger than the size of the [memory] block. *Id.* at 5:59-61. The "size of the sector of the memory card is stored in the registers of the memory as well." *Id.* at 5:66-67. According to the RE486 Patent, this enables the MMC memory card "to use the register READ_BL_LEN indicating the [memory] block size" to calculate the capacity of the memory card. *Id.* at 6:1-4. In this embodiment, the significance of the parameters of the formula used to calculate memory card capacity for a memory card according to the MMC specification changes. *Id.* at 6:14-20. Particularly, "the parameter C_SIZE has been change[d] so that it signifies kilobytes instead of bytes." *Id.* at 6:21-23. Accordingly, the maximum memory capacity is 4 terabytes. *Id.* at 6:24-33.

In another embodiment, the increase of the memory capacity of the memory card is implemented by using additional values for the parameter READ_BL_LEN to increase BLOCK_LEN, upon which the memory card capacity can be calculated. Ex. 1001 at 7:50-54. The RE486 Patent discloses using additional values (12-15) for the parameter READ_BL_LEN, such that the maximum memory capacity is 64 gigabytes. *Id.* at 7:54-60.

In a third embodiment, "the number of address bits is increased" to increase the memory capacity of a memory card. *Id.* at 8:42-43. This embodiment is implemented "preferably by doubling the number of address bits from, for example, 32 bits to 64 bits." *Id.* at 8:43-44. In this embodiment, values 12-15 are taken into use in the parameter READ_BL_LEN as described above. *Id.* at 8:40-42. In this embodiment, the value of parameter READ_BL_LEN is used to calculate the maximum memory capacity and in determining the size of the memory blocks. *Id.* at 8:50-53. In this third embodiment, "the increase in the number of address bits can be implemented by several means." *Id.* at 9:5-6. One option is that "a special command is specified, which indicates to the memory card that it is an expanded address." *Id.* at 9:7-8. This "type of special command can be implemented in the present command register CSD or in the expanded command register EXT_CSD." *Id.* at 9:11-14. Another possibility is to use a

switch- command, wherein "the parameter of the switch command [...] indicates which command is in question at a certain time." *Id.* at 9:14-17.

B. Prosecution History of the RE486 Patent

The RE486 Patent (U.S. Reissue App. No. 13/902,258) was filed on May 24, 2013, seeking reissue of U.S. Patent 7,257,669 ("the '669 Patent"), and issued on April 2, 2015. Ex. 1001. The RE486 Patent claims the benefit of a foreign application and a cancelled parent application, the earliest filed on February 7, 2003. Ex. 1009 at ¶¶108-109, 116.

During prosecution of the '669 Patent, the applicant argued that "[m]emory capacity and an addressing method used (such as byte addressing or sector addressing) are not the same nor are they equivalent to each other" and that "[t]he addressing method is not necessarily dependent on the memory capacity of the card" to overcome a rejection based on Applicant Admitted Prior Art. Ex. 1004 at 39-40.

The Preliminary Amendment filed with the Reissue application of the RE486 Patent sought to cancel claims 2 and 7, amend claims 6 and 8, and add claim 22. Ex. 1002 at 299-306. Claims 2 and 7 had claimed, "at least one of the following: addressing two or more memory locations with one address; increasing the number bits that can be used in an address." Ex. 1003 at 10:19-25, 10:62-67. During prosecution of the RE486 Patent, the applicant argued that prior art cited by

the Examiner "improperly equates a 'card address' in the RCE register with the claimed 'addressing data being indicative of at least one addressing method supported,' as recited in claims 1, 4, 6, 12, 13, 10 and 31." Ex. 1002 at 65. The applicant failed to define "an addressing method," but provided a "non-limiting example" of two addressing methods: an "expanded addressing method" and a "basic addressing method." *Id*.

VI. CLAIM CONSTRUCTION (37 C.F.R. § 42.104(b)(3))

The Patent Office has adopted a rule by which claims are construed in accordance with "the standard used in federal courts, in other words, the claim construction standard that would be used to construe the claim in a civil action under 35 U.S.C. 282(b), which is articulated in *Phillips v. AWH Corp*., 415 F.3d 1303 (Fed. Cir. 2005)." 83 FR 51340. Under this standard, claim construction begins with the language of the claims. *Phillips*, 415 F.3d at 1312-14. The "words of a claim are generally given their ordinary and customary meaning," which is "the meaning that the term would have to a person of ordinary skill in the art in question at the time of the invention, i.e., as of the effective filing date of the patent application." *Id*. at 1312-13. The specification is "the single best guide to the meaning of a disputed term and . . . acts as a dictionary when it expressly defines terms used in the claims or when it defines terms by implication." *Id*. at 1321

(internal quotation marks omitted). The prosecution history is another source of intrinsic evidence. *Id*. at 1317.

All claim terms of challenged claims of the RE486 Patent have been accorded their plain and ordinary meaning as understood by person of ordinary skill in the art and consistent with the intrinsic record. Petitioner's interpretation of the claim terms in the RE486 Patent is further explained for each claim limitation in relation to the prior art discussed in the proposed grounds for invalidity, below, in Grounds 1-4.

Under the *Phillips* standard and for clarity, Petitioner provides the following specific constructions.[1]

A. "addressing data"

Challenged claims recite "addressing data." A POSITA would have recognized that this phrase means "data indicative of an addressing method." Ex. 1009 at ¶128. "[A]ddressing data" is not a term of art. Ex. 1009 at ¶129. The specification of the RE486 Patent describes one embodiment in which an indication "that the memory card functions in a sector-based manner" is "stored

[1] Petitioner reserves the right to address any claim construction positions taken by the Patent Owner in its Preliminary Response, if any, including under 37 C.F.R. § 42.108(c). Petitioner further reserves its ability to show that claims of the RE486 Patent are invalid under 35 U.S.C. §112 in the co-pending litigation, despite offering explicit and implicit claim constructions herein.

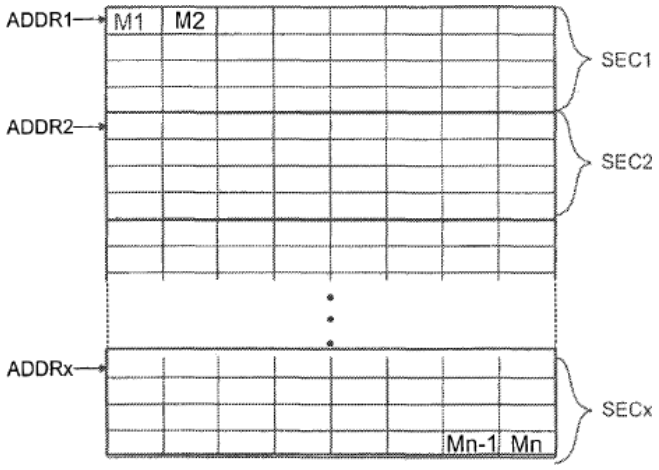
preferably in one bit" and constitutes "addressing data." Ex. 1001 at 5:51-54; Ex. 1004 at 38-41; Ex. 1009 at ¶129. The specification of the RE486 Patent further describes that the "value of the addressing data is stored to the memory card advantageously in the manufacturing phase of the memory card." Ex. 1001 at 5:57-59.

Accordingly, a POSITA would have recognized that "addressing data" is "data indicative of an addressing method." Ex. 1009 at ¶130.

B. "an addressing method"

Challenged claims recite "an addressing method." A POSITA would have recognized this phrase means "a technique of accessing data by way of its location." Ex. 1009 at ¶131.

"[A]n addressing method" is not a term of art. *Id.* at ¶132. The specification explains that the "aim of the present invention to provide an improved addressing method for addressing memory locations." Ex. 1001 at 3:3-4. The patent states: "FIG. 2 shows an addressing method[.]" *Id.* at 3:55-56. Figure 2 of the RE486 Patent is shown below:



The specification describes three embodiments. *Id.* at 1:41-46, 7:49- 55, 8:37-49. During prosecution of the '669 Patent, the applicant argued that "[t]he addressing method may be byte addressing or sector addressing." Ex. 1004 at 39. The Applicant asserted that "[t]he invention is a usage of a sector address instead of a byte address and, more particularly, adapting to the type of addressing modes supported (for example, such as byte or sector addressing) as indicated by the memory device." *Id.* Further, the Applicant argued that "[m]emory capacity and an addressing method used (such as byte addressing or sector addressing) are not the same nor are they equivalent to each other." *Id.*

During prosecution of the RE486 Patent, the applicant argued that "[a]s a non-limiting example from the '669 patent specification, two different supported addressing methods may include an 'expanded addressing' method and a 'basic addressing' method." Ex. 1002 at 65 (citing the '669 Patent at 9:39-58).

Accordingly, a POSITA would have recognized that "an addressing method" is "a technique of accessing data by way of its location." Ex. 1009 at ¶135.

C. "memory location"

Challenged claims recite a "memory location." A POSITA would have recognized that this means "a location in memory where data is stored." Ex. 1009 at ¶136.

The specification of the RE486 Patent describes semiconductor memory, "where there are several memory locations that can be addressed." Ex. 1001 at 1:60-62. The specification further explains that a memory location is not limited to a particular size, stating: "[e]ach memory location typically comprises a specific number of bits, such as 8 bits (a byte), 16 bits (a word), 32 bits (a double-word), or even 64 bits." *Id.* at 1:62-65. Accordingly, "the amount of data that can be addressed with one piece of address data is the amount of bits in the memory location in question." *Id.* at 1:65-67.

Accordingly, a POSITA would have recognized that "memory location" is "a location in memory where data is stored." Ex. 1009 at ¶138.

VII. LEVEL OF ORDINARY SKILL IN THE ART

A person of ordinary skill in the art ("POSITA") in the field of developing the technology of the RE486 Patent would have a B.S. degree in electrical engineering or a related field, in combination with two to three years training in

memory system design. Ex. 1009 at ¶71. This description is approximate, and a higher level of training might make up for less education, and vice-versa. *Id.*

VIII. BRIEF DESCRIPTION OF PRIOR ART RELIED UPON

A. United States Patent No. 6,279,114 to Toombs et al.

Toombs describes a memory card, such as a MultiMedia Card ("MMC"), comprising several memory locations for storing data. Ex. 1005 at 10:22-34, FIG. 17A; Ex. 1009 at ¶¶140-49. Toombs describes a bit stored to a register on the memory card that indicates whether partial data blocks can be read from card memory. Ex. 1005 at 11:10-13. If partial block reads are enabled, data blocks as small as a single byte may be read by the host. *Id.* at 11:17-18.

Toombs further describes a bit stored to a register on the memory card that indicates whether partial data blocks can be written by the host. *Id.* at 12:45-47. Toombs therefor discloses two independent examples of a basic addressing method and an expanded addressing method.

B. United States Patent No. 6,314,504 to Dent

Dent describes a processor architecture and associated method to improve the efficiency of memory accesses and thereby reduce power consumption. Ex. 1006 at Abstract; Ex. 1009 at ¶¶151-52. Index registers are employed in calculating an effective address for memory-reference instructions. Ex. 1006 at 6:1-3. Each index register comprises a mode byte indicating how the register value

shall be used. *Id.* at 6:3-5. Depending on the value of the mode byte, the remaining address bits can address up to 2GB of memory or up to 4GB of memory. Therefore, Dent describes a basic and an expanded addressing method wherein the expanded addressing method enables addressing in a higher number of memory locations than the basic addressing method. Ex. 1009 at ¶¶152-55.

C. The PCMCIA Standard

The PCMCIA standard defines specifications for PC Cards and communications between PC Card and their hosts. Ex. 1007 at 1; Ex. 1009 at ¶157. The 26 address signals at the PC Card connector can directly address only 64 MBytes of memory. Ex. 1007 at 61. However, a Configuration Option register can provide six address extension bits which, when combined with the PC Cards 26 address signals (A[25::0]), allow for the addressing of 4 Gigabytes. *Id.* at 56. The selection of the Common Memory Address Extension field option is specified by a tuple contained in the memory card's Card Information Structure. *Id.* at 15, 56. PCMCIA thus describes a basic addressing method, wherein the card can address 64MB of memory, and an expanded addressing method, wherein the card uses the Configuration Option Register to employ an additional six address bits to address 4GB of memory. *Id.*; Ex. 1009 at ¶¶159-61.

The "Wayback Machine" maintained by Internet Archive at archive.org confirms that the PCMCIA Standard was available from the Personal Computer

Memory Card International Association (PCMCIA) website as of at least November 2001. Ex. 1007 at 2; Ex. 1015 at 17-26. The PCMCIA Standard (Release 7.0) was first archived by the Wayback Machine on November 28, 2001. The PC Card website homepage included a link presented on a tab entitled "About PCMCIA." Ex. 1015 at 18-19. The linked page was entitled "About PCMCIA" and included a link to "PC Card Standards." *Id.* at 20-24. The linked page was entitled "Detailed Overview of the PC Card Standard" and included the PCMCIA/PC Card Standard Release History, listing "PC Card Standard 7.0 Release" in February 1999. *Id.* at 21. The previous linked page entitled "About PCMCIA" included another link directing users to a page where PC Card Standard could be "ordered through this site." *Id.* at 18-19. This linked page was entitled "Online Order Form" and enables purchase of the updated standard by both members (for $50) and non-members (for $299). *Id.* at 25-26.

D. Revision 3a of the ATA-6 Standard

Revision 3a of the ATA-6 Standard specifies the AT Attachment Interface between host systems and storage devices. Ex. 1008 at 15; Ex. 1009 at ¶163. This interface is used by CompactFlash storage devices. Ex. 1008 at 17. Revision 3a of the ATA-6 Standard describes a 48-bit address feature in addition to the standard 28-bit addressing. *Id.* at 109-110. The standard 28-bit addressing method is limited to addressing 137GB of memory. *Id.*; Ex. 1009 at ¶166. The optional 48-

bit Address feature set allows device capacity of up to approximately 144 petabytes. Ex. 1008 at 109-110. A bit stored in a register on the storage device indicates whether 48-bit addressing is supported. *Id.* at 78. Revision 3a of the ATA-6 Standard thus describes a basic addressing method, wherein the card uses 28-bit addressing to address 137GB of memory, and an expanded addressing method, wherein the card uses 48-bit addressing to address 144 petabytes of memory. *Id.*; Ex. 1009 at ¶166.

The "Wayback Machine" maintained by Internet Archive at archive.org confirms that Revision 3a of the ATA-6 Standard was publicly available online from the T13 website as of at least January 2002. Ex. 1008; Ex. 1015 at 1-2. Revision 3a of the ATA-6 Standard was first archived by the Wayback Machine on January 23, 2002. T13 is a Technical Committee of Accreted Standards Committee NCITS. Ex. 1015 at 4. The T13 homepage included a list of project drafts created or maintained by T13, including a PDF link to "ATA/ATAPI - 6 revision 3a." *Id.* at 4-16. The PDF document was the revision 3a of the ATA/ATAPI - 6 standard, and bears the revision date "14 December 2001" on the cover page. *Id.* at 28. This PDF document was available for free download as of at least January 2002. *Id.* at 12.

Further evidence that Revision 3a of the ATA-6 Standard was publicly available before the priority data of the RE486 Patent is shown by U.S. Patent

Publication 2003/0235408 to Silvester et al. ("the '408 Publication"), which incorporates Revision 3a of the ATA-6 Standard by reference. Ex. 1014 at ¶[0023]; Ex. 1009 ¶165.

IX. CLAIM-BY-CLAIM EXPLANATION OF GROUNDS OF UNPATENTABILITY

All of the Challenged Claims are unpatentable as explained below.

A. GROUND 1: Claims 6, 8-11, 25-27, and 30-31 are anticipated by Toombs

1. Independent Claim 6

a. 6 (preamble) "A memory card comprising:"

To the extent the preamble is a limitation, the preamble is taught by Toombs. Ex. 1009 at ¶168. Toombs describes "the architecture of a MultiMediaCard card of a preferred embodiment according to the present invention." Ex. 1005 at FIG. 14.

Accordingly, Toombs discloses this claim limitation. Ex. 1009 at ¶169.

b. 6(a) "several memory locations for storing data, and in which memory card is stored at least one parameter, the memory card configured so that the number of memory locations of the memory card can be calculated on the basis of said at least one parameter"

As discussed in *Section VI.C.*, a POSITA would have recognized that a memory location is "a location in memory where data is stored." Ex. 1009 at ¶¶137-38.

Toombs discloses several memory locations for storing data, such as the Card-Specific Data (CSD) register. Ex. 1009 ¶170. The CSD register stores parameters regarding the card. Ex. 1005 at 10:22-34, FIG. 17A. One value stored in the CSD register is the parameter C_SIZE. *Id.* at 11:45; Ex. 1009 at ¶¶171-172. In Toombs, the capacity of the card can be calculated from the parameter C_SIZE, as shown below. Ex. 1005 at 11: 50-55.

memory capacity = BLOCKNR * BLOCK_LEN
where
BLOCKNR = (C_SIZE+1) * MULT
MULT = $2^{C\_SIZE\_MULT+2}$ (C_SIZE_MULT < 8)
BLOCK_LEN = $2^{READ\_BL\_LEN}$ (READ_BL_LEN < 12)

The memory block length is multiplied by the number of memory blocks, which is a function of C_SIZE, to calculate the capacity of the memory card. *Id.* at 11:45-60. Thus, the calculation of the number of memory locations/capacity as described in Toombs is derived from the parameter C_SIZE. Accordingly, Toombs' disclosure of the parameter C_SIZE teaches this element. Ex. 1009 ¶¶170-74.

c. 6(b) "the memory card configured so that a specific number of bits is reserved for said at least one parameter, and"

Toombs discloses reserving 12 bits (bits 62-73) of the CSD register for the C_SIZE parameter:

NAME	FIELD	WIDTH	CELL TYPE	CSD-SLICE
CSD STRUCTURE	CSD_STRUCTURE	2	R	[127:126]
MMC PROTOCOL VERSION	MMC_PROT	4	R	[125:122]
RESERVED	-	2	R	[121:120]
DATA READ ACCESS-TIME-1	TAAC	8	R	[119:112]
DATA READ ACCESS-TIME-2 IN CLK CYCLES (NSAC*100)	NSAC	8	R	[111:104]
MAX. DATA TRANSFER RATE	TRAN_SPEED	8	R	[103:96]
CARD COMMAND CLASSES	CCC	12	R	[95:84]
MAX. READ DATA BLOCK LENGTH	READ_BL_LEN	4	R	[83:80]
PARTIAL BLOCKS FOR READ ALLOWED	READ_BL_PARTIAL	1	R	[79:79]
WRITE BLOCK MISALIGNMENT	WRITE_BLK_MISALIGN	1	R	[78:78]
READ BLOCK MISALIGNMENT	READ_BLK_MISALIGN	1	R	[77:77]
DSR IMPLEMENTED	DSR_IMP	1	R	[76:76]
EXTERNAL V_{PP}	VPROG	2	R	[75:74]
DEVICE SIZE MANTISSA	C_SIZE_MANT	8	R	[73:66]
DEVICE SIZE EXPONENT	C_SIZE_EXP	4	R	[65:62]
MAX. READ CURRENT @V_{DD} MIN	VDD_R_CURR_MIN	3	R	[61:59]
MAX. READ CURRENT @V_{DD} MAX	VDD_R_CURR_MAX	3	R	[58:56]

FIG. 17A

Ex. 1005 at FIG. 17A (annotated).

A POSITA would understand that C_SIZE is a parameter expressed in terms of a mantissa and an exponent. Ex. 1009 at ¶175. Together, the 8-bit mantissa (C_SIZE_MANT) and the 4-bit exponent (C_SIZE_EXP) represent the 12-bit parameter C_SIZE. *Id.* Toombs' disclosure of reserving 12 bits of the CSD register for the C_SIZE parameter discloses "a specific number of bits is reserved for said at least one parameter." *Id*.

Accordingly, Toombs discloses this claim limitation. *Id*.

d. 6(c) "the memory card further configured to have stored therein an addressing data, said addressing data being indicative of at least one addressing method supported"

Toombs discloses commands for transferring data. Ex. 1005 at 27:30-32. The basic unit of data transfer between the MultiMediaCard system is one byte. *Id.* at 27:28-29. Some transfer commands involve blocks of data, in which case the block size/length is defined as an integer multiple of bytes. *Id.* at 27:30-32, 43-44. The size of the block is the number of bytes which will be transferred when one data block is sent (or received) by the host. *Id.* at 27:44-46. The size of a block is either programmable or fixed, and information about the block sizes and the programmability is stored in the CSD. *Id.* at 27:47-49.

The maximum length for a data block is computed as $2^{READ\_BL\_LEN}$, where READ_BL_LEN is a value between 0-11. Ex. 1010 at 55. The block length can therefore be in the range 1, 2, 4...2048 bytes. Ex. 1005 at 11:5-9. The MMC Specification expressly describes that the data block length can be 512 bytes, where READ_BL_LEN=9 (2^9=512). Ex. 1010 at 55, 85. In the following discussion, a nominal data block length of 512 bytes is used. Ex. 1009 at ¶178.

The memory storage of the card described in Toombs is divided into physical memory blocks. Ex. 1005 at 27:37-42. The size of each physical block of the card is defined by the READ_BL_LEN field of the CSD as well. Ex. 1010 at

27. Accordingly, the maximum *data* block that can be transferred with a single block command is equal to the size of a *physical* memory block of the card. Ex. 1009 at ¶179.

Toombs describes a single block read command (READ_SINGLE_BLOCK) referred to as CMD17. Ex. 1005 at 20:16-29, Fig. 39. This command reads a data block of the size selected by the SET_BLOCKLEN command (CMD16). *Id.* at Fig. 39. The default block length is specified in the CSD. *Id*.

The CSD further includes a READ_BL_PARTIAL field that indicates whether partial data blocks can be read with block read commands, such as CMD17. *Id.* at 11:10-13, FIG 17A. When the READ_BL_PARTIAL is not enabled, only data blocks of the size READ_BL_LEN can be used for block-oriented data transfers. *Id.* at 11:13-15. Accordingly, when the READ_BL_PARTIAL field is set to "0," only data blocks of 512 bytes can be read. *Id.* But, when READ_BL_PARTIAL is enabled, smaller blocks can be read by the host as well. *Id.* at 11:15-17; Ex. 1009 at ¶181. Thus, when READ_BL_PARTIAL is set to "1," data blocks smaller than the 512 byte block size can be read. *Id.* at 11:15-17; Ex. 1009 ¶181.

The size of a partial data block is set by the SET_BLOCKLEN command (CMD16), which specifies the smaller block size for subsequent commands. Ex. 1005 at 19:23-24, FIG. 39. In one example, CMD16 can be used to set

SET_BLOCKLEN to 128 bytes, equivalent to 1/4 of the data block length defined by READ_BL_LEN discussed above. Ex. 1009 at ¶182. Alternatively, SET_BLOCKLEN may be 1 byte, equivalent to 1/512 of the data block length. *Id*.; Ex. 1005 at 12:48-49. When read block partial is not enabled (READ_BL_PARTIAL=0), CMD16 will not allow the host to define block lengths smaller than READ_BL_LEN. Ex. 1009 at ¶182. In this case, the value set by SET_BLOCKLEN must be identical to READ_BL_LEN. *Id*.

Returning to the single block read command (CMD17) described in Toombs, the host is restricted to reading data blocks of length READ_BL_LEN, *e.g.*, 512 bytes, where READ_BL_PARTIAL is not enabled. *Id.* at 11:13-15. However, if read block partial is enabled, smaller blocks whose starting and ending address are entirely contained within one physical memory block, may also be transmitted. Ex. 1005 at 20:10-13. In this case, the single block read command (CMD17) reads a block of the size selected by the SET_BLOCKLEN (CMD16), which can be as small as one byte. *Id.* at 11:15-17, Fig. 39; Ex. 1009 ¶183.

Accordingly, the bit indicating whether read block partial is enabled or not enabled is indicative of an addressing method because it indicates whether the memory card system can address data blocks of length READ_BL_LEN or data blocks smaller than READ_BL_LEN using CMD16. Ex. 1009 at ¶184. Toombs therefore discloses this claim element. *Id*.

Toombs also discloses a second example of this claim element. Ex. 1009 ¶185. Toombs describes a stream write command, CMD20, that writes a stream of data to the card, beginning with a host-supplied starting address, and ending when the host issues a stop command. Ex. 1005 at 20:38-52.

The CSD of the card described in Toombs further includes a WRITE_BL_PARTIAL field that indicates whether partial data blocks can be written. *Id.* at 12:42-45, 20:44-48. When WRITE_BL_PARTIAL is set "the data stream can start and stop at any address within the card address space." *Id.* at 20:44-47. But, when WRITE_BL_PARTIAL is not enabled (set to "0"), "the data transfer starts and stops only at block boundaries." Ex. 1009 at ¶189. Thus, as explained in Toombs:

> If a stream write operation is stopped prior to reaching the block boundary and partial block data transfer is allowed (as defined in the CSD), the part of the last block will be packed as a partial block and programmed. If partial blocks are not allowed, the remaining data will be discarded.

Id. at 14:49-53.

Accordingly, the bit indicating whether write block partial is enabled or not is addressing data indicative of an addressing method because it indicates whether the memory card system can address data "at any address within the card" or is limited to addressing data transfers to "start[] and stop[] only at [physical] block

boundaries." *Id.* 20:44-47; Ex. 1009 at ¶188. Toombs therefore discloses another example of this claim element. *Id.*

e. 6(d) "wherein the addressing data indicates either a basic addressing method or an expanded addressing method, and"

As described in *Section IX.A.1.d*, Toombs discloses a bit stored to the CSD register that indicates whether partial block reads are permitted.

In one addressing method, the memory card system does not permit partial block reads (READ_BL_PARTIAL=0), and the host can only read data blocks of length READ_BL_LEN. Ex. 1009 at ¶190-91. This is an expanded addressing method. *Id*. In the example discussed above, the data block length defined by READ_BL_LEN of is 512 bytes. *Id.* When read block partial is not enabled, CMD16 must be identical to READ_BL_LEN. *Id.* Thus, in the expanded addressing method described in Toombs, the smallest data block that can be read is READ_BL_LEN, *e.g.*, 512 bytes. *Id*.

In another addressing method, the memory card system permits partial block reads (READ_BL_PARTIAL=1), and the host can read smaller data blocks of length SET_BLOCKLEN using CMD16. Ex. 1009 at ¶192. This is a basic addressing method. *Id.* When read block partial is enabled, CMD16 can be used by the host to access blocks smaller than READ_BL_LEN. *Id*.; Ex. 1005 at 11:10-18. In this basic addressing method, the host can set the value of

SET_BLOCKLEN using CMD16, such that a single byte can be read with a block read command. Ex. 1009 at ¶192.

Toombs discloses a second example of this claim limitation. *Id.* at ¶¶193-95. As described in Section IX.*A.1.d*, the CSD register stores a bit that indicates whether partial block writes are permitted. This bit indicates whether the memory card system can initiate a stream write "at any address within the card" or is limited to initiating stream writes that "start[] and stop[] only at [physical] block boundaries." Ex. 1005 at 20:44-47, Ex. 1009 at ¶194. In this example, the "basic" method corresponds to the case where the partial writes are not permitted, and the start addresses (and stop addresses) are limited to the physical block boundaries. *Id.* The "expanded" method corresponds to when partial writes are permitted, allowing the host to use "any address within the card" as the start and stop points for the transfer. *Id*.

Toombs therefore discloses two examples of this claim limitation. Ex. 1009 at ¶195.

f. 6(e) "wherein the expanded addressing method enables the addressing of data in a larger number of memory locations than the basic addressing method."

As discussed in *Section IX.A.1.e*, Toombs discloses addressing data indicative of either a basic addressing method or an expanded addressing method

with a single bit that either enables or prohibits partial block reads. Ex. 1009 at ¶196.

Toombs discloses an expanded addressing method indicated by READ_BL_PARTIAL = 0 in which block lengths of READ_BL_LEN are used for block oriented data transfers. *Id.* at ¶197. As an example, the read block length may be set in the CSD register to be 512 bytes. *Id.* Accordingly, the expanded addressing method described in Toombs enables the addressing of data in each memory location of the 512 byte block. *Id*.

Toombs also discloses a basic addressing method indicated by READ_BL_PARITAL=1 in which the host can access subunits of READ_BL_LEN by using CMD16 to set SET_BLOCKLEN for block-oriented commands. *Id.* at ¶198. For example, CMD16 can be used to set SET_BLOCKLEN as small as a single byte. *Id*.; Ex. 1005 at 11:17-18. This basic addressing method enables the addressing of data in a single memory location: the 1 byte subunit of the 512 byte READ_BL_LEN. Ex. 1009 at ¶198.

Accordingly, in this example in Toombs, the "expanded" addressing method enables addressing of data in a larger number of locations (512 byte locations) compared to the "basic" method, which enables addressing of data in a smaller number of locations (e.g., a single byte location). Ex. 1009 at ¶199. Toombs therefore discloses this claim limitation. *Id*.

Toombs also discloses a second example of this claim element. *Id.* at ¶200. As discussed in the previous claim element, write block partial being enabled (WRITE_BL_PARTIAL=1), corresponds to the "expanded" addressing method. In this mode, the host can invoke a stream write command and designate "any address" in the card at which to begin writing data, as opposed to the "basic" method, where the host is restricted to starting a write only at the physical block boundaries of the card. *Id.* Accordingly, this "expanded" method enables the host to address a larger number of memory locations ("any address") compared to the basic addressing method (only the relatively smaller number of addresses corresponding to physical block boundaries are permitted). *Id*.

Toombs therefore discloses two examples of this claim limitation. Ex. 1009 at ¶¶196-201.

Because Toombs discloses each and every element of Claim 6, Toombs anticipates Claim 6. *Id.* at ¶202.

2. Dependent Claim 8 "The memory card according to claim 6, comprising a bus connection block for connecting the memory card to a device and for transferring data between the device and the memory card."

Toombs describes that the MultiMediaCard bus connects the MultiMediaCard host to a MultiMediaCard card comprising an I/O device. Ex. 1005 at 6:61-65. FIG. 4 of Toombs shows a MultiMediaCard bus system:

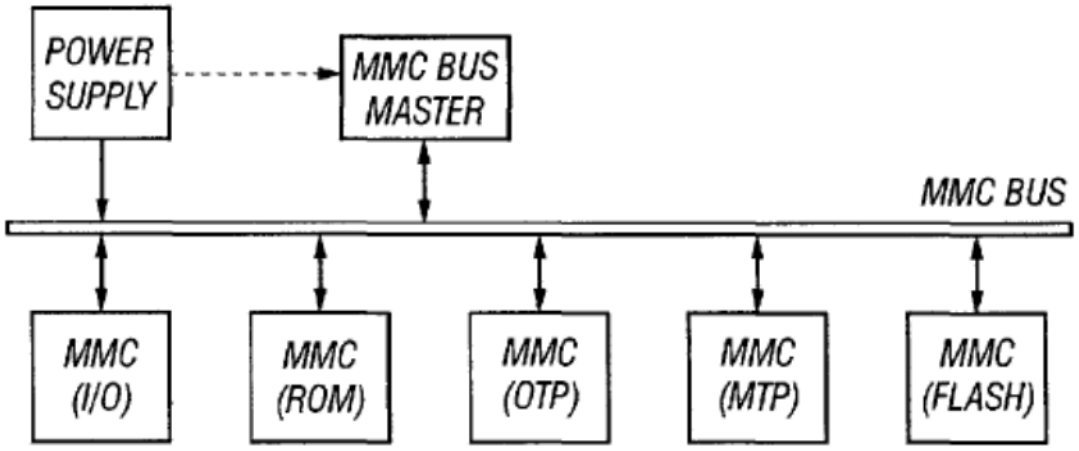


FIG. 4

Toombs describes that [i]n this MultiMediaCard bus protocol, the payload data transfer between the host and the cards is specifically designed to be bidirectional so that data can be transferred between the host to the cards." *Id.* at 7:14-17.

The memory card interfaces with the MMC Interface Controller, as shown in FIG. 14 of Toombs:

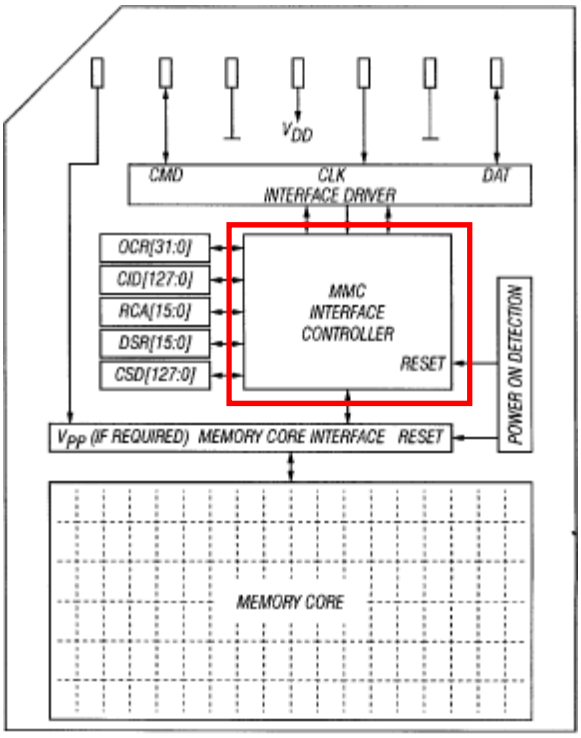


FIG. 14

Toombs therefore discloses a bus connection block for connecting the memory card to a device and for transferring data between the device and the memory card. Ex. 1009 ¶¶203-06.

Toombs therefore discloses this claim. Ex. 1009 at ¶207.

3. Dependent Claim 9 "The memory card according to claim 6, wherein data is arranged to be stored and read in the memory card block-by-block."

Toombs describes that the memory card system uses sequential command and block-oriented commands. Ex. 1005 at 8:32-34. Both read and write operations allow either single or multiple block transmission. *Id.* at 8:46-48.

For example, Toombs discloses a write block command, CMD24, which writes a block of the size selected by the SET_BLOCKLEN command. *Id.* at FIG. 41 (below); Ex. 1009 at ¶209.

CMD INDEX	TYPE	ARGUMENT	RESP	ABBREVIATION	COMMAND DESCRIPTION
CMD24	ADTC	[31:0] DATA ADDRESS	R1B	WRITE_BLOCK	WRITES A BLOCK OF THE SIZE SELECTED BY THE SET_BLOCKLEN COMMAND.[1]
CMD25	ADTC	[31:0] DATA ADDRESS	R1B	WRITE_MULTIPLE_ BLOCK	CONTINUOUSLY WRITES BLOCKS OF DATA UNTIL A STOP_TRANSMISSION FOLLOWS.
CMD26	ADTC	[31:0] STUFF BITS	R1B	PROGRAM_CID	PROGRAMMING OF THE CARD IDENTIFICATION REGISTER. THIS COMMAND SHALL BE ISSUED ONLY ONCE PER MMC CARD. THE CARD CONTAINS HARDWARE TO PREVENT THIS OPERATION AFTER THE FIRST PROGRAMMING. NORMALLY THIS COMMAND IS RESERVED FOR THE MANUFACTURER.
CMD27	ADTC	[31:0] STUFF BITS	R1B	PROGRAM_CSD	PROGRAMMING OF THE PROGRAMMABLE BITS OF THE CSD.

FIG. 41

Toombs describes that WRITE_BL_LEN is used to indicate the block length for write operations. Ex. 1005 at 12:38-40. When write block partial is not enabled, only the WRITE_BL_LEN block size can be used for block-oriented data writing. *Id.* at 12:45-47. Toombs similarly describes block-oriented commands that enable data to be read out of the card block-by-block. Ex. 1009 at ¶¶210-13.

Toombs therefore discloses this claim. *Id.* at ¶213.

4. Dependent Claim 10 "The memory card according to claim 9, wherein the memory locations of one block are arranged to be addressed with one address."

Toombs discloses a write block command, CMD24. Ex. 1005 at FIG. 41. The write block command (CMD24) functions to write a block of the size selected by the SET_BLOCKLEN command. *Id.* The WRITE_BL_LEN of the preferred embodiment according to Toombs is used to indicate the block length for write operations. *Id.* at 12:38−41. The card receives one address as the argument, and

programs the block of data into a location of memory starting with the address received as the command argument. *Id.* at 22:11-15, FIG. 41; Ex. 1009 at ¶214.

Toombs similarly describes block-oriented commands that enable multiple blocks of data to be read using a single starting address. Ex. 1009 at ¶215. For example, CMD17 can be used to read a single data block from the card memory using the starting address as the argument. Ex. 1005 at FIG. 39.

Toombs therefore discloses this claim. *Id.* at ¶¶215-217.

5. Dependent Claim 11 "The memory card according to claim 6, wherein the memory card is a memory card according to MultiMediaCard specifications."

Toombs describes a MultiMediaCard in accordance with the specification, including the use of a MMC bus, "Serial Bus (MMC)," as well as various "MMC Adapters" to connect the host devices to MMC cards. Ex. 1009 at ¶¶217-219; Ex. 1005 at FIG. 1.

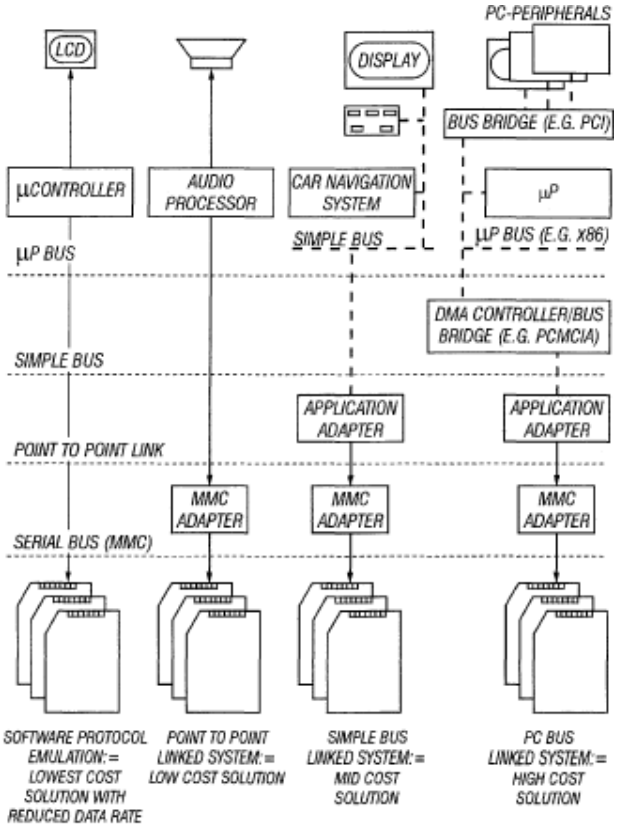


FIG. 1

6. Dependent Claim 25 "The memory card according to claim 6, wherein the memory card is configured so that, if the addressing data indicates that the memory card supports the expanded addressing method, the memory card uses the expanded addressing method in response to a successful reading of the addressing data that indicates support of the expanded addressing method."

As discussed in *Section VIII.A*, Toombs describes an expanded addressing method indicated by the READ_BL_PARTIAL setting. Ex. 1009 at ¶221. Toombs explains that in the setup procedure, command CMD9 is used to read the contents of the CSD register to obtain, e.g., block length, card storage capacity, etc. Ex. 1005 at 18:42-45, 24:34-35. When the host reads a "0" in the

READ_BL_PARTIAL field of the CSD register, it will use the expanded addressing method. *Id*.; Ex. 1009 ¶221.

Moreover, as discussed in *Section VIII.A*, Toombs describes an expanded addressed method indicated by WRITE_BL_PARTIAL=1 (*i.e.,* partial writes are supported) in the CSD. When a host reads a "1" in the WRITE_BL_PARTIAL field of the CSD register, it can use the expanded addressing method, and invoke stream writing without regard to whether the start and stop points in the card address space correspond to physical block boundaries. Ex. 1005 at 20:44-47; Ex. 1009 at ¶222.

A POSITA would understand that CMD9 must be used before any addressing method can be used. Ex. 1009 at ¶223.

7. Dependent Claim 26 "The memory card according to claim 6, further comprising a register for storing the addressing data."

Toombs discloses a Card-Specific Data register (CSD) "responsible for providing information to the MultiMediaCard host on how to access the card content." Ex. 1005 at 10:24-25. Specifically, "the CSD register stores values defining the data format. *Id.* at 10:22-33. One value stored in the CSD register is the bit that indicates whether read block partial is enabled. *Id.* at FIG. 17A; Ex. 1009 at ¶226. In another example disclosed in Toombs, the CSD stores a bit that indicates whether write block partial is enabled. Ex. 1005 at FIG. 17B; Ex. 1009 at

¶236. Each of these bits are independently "addressing data." Ex. 1009 at ¶¶226-27. Toombs therefore discloses this claim. *Id.* at ¶229.

8. Dependent Claim 27 "The memory card according to claim 26, wherein the stored addressing data comprises one bit."

Both READ_BL_PARTIAL and WRITE_BL_PARTIAL, which each independently constitute the stored addressing data, are single-bit settings in the CSD register (i.e., "Width = 1"). Ex. 1009 at ¶¶230-31. Excerpts from Figures 17A and 17B of Toombs illustrate this:

NAME	FIELD	WIDTH	CELL TYPE	CSD-SLICE
PARTIAL BLOCKS FOR READ ALLOWED	READ_BL_PARTIAL	1	R	[79:79]
PARTIAL BLOCKS FOR WRITE ALLOWED	WRITE_BL_PARTIAL	1	R	[21:21]

Toombs therefore discloses this claim limitation. Ex. 1009 at ¶231.

9. Independent Claim 30

a. 30(pre) "A memory card comprising:"

As shown above in *Section IX.A.1.a*, Toombs teaches this claim element. Ex. 1009 at ¶¶168-69.

b. 30(a) "several memory locations for storing data, and in which memory card is stored at least one parameter, the memory card configured so that the number of memory locations of the memory card can be calculated on the basis of said at least one parameter,"

As shown above in *Section IX.A.1.b*, Toombs teaches this claim element.

Ex. 1009 at ¶¶170-74.

c. 30(b) "the memory card configured so that a specific number of bits is reserved for said at least one parameter, and"

As shown above in *Section IX.A.1.c*, Toombs teaches this claim element.

Ex. 1009 at ¶¶175-76.

d. 30(c) "the memory card further configured to have stored therein an addressing data, said addressing data being indicative of at least one addressing method supported,"

As shown above in *Section IX.A.1.d*, Toombs teaches this claim element.

Ex. 1009 at ¶¶177-89.

e. 30(d) "wherein the addressing data indicates either a basic addressing method or an expanded addressing method,"

As shown above in *Section IX.A.1.e*, Toombs teaches this claim element.

Ex. 1009 at ¶¶190-95.

f. 30(e) "wherein the expanded addressing method enables the addressing of data in a larger number of memory locations than the basic addressing method, and"

As shown above in *Section IX.A.1.f*, Toombs teaches this claim element. Ex. 1009 at ¶¶196-202.

g. 30(f) "wherein the memory card is configured so that, if the addressing data indicates that the memory card supports the expanded addressing method, the memory card uses the expanded addressing method in response to a successful reading of the addressing data that indicates support of the expanded addressing method."

As shown above in *Section IX.A.6*, Toombs teaches this claim element. Ex. 1009 at ¶¶221-25.

10. Independent Claim 31

a. 31(pre) "A memory card comprising:"

As shown above in *Section IX.A.1.a*, Toombs teaches this claim limitation. Ex. 1009 at ¶¶168-69.

b. 31(a) "several memory locations for storing data, and in which memory card is stored at least one parameter, the memory card configured so that the number of memory locations of the memory card can be calculated on the basis of said at least one parameter,"

As shown above in *Section IX.A.1.b*, Toombs teaches this claim limitation. Ex. 1009 at ¶¶170-74.

c. 31(b) "the memory card configured so that a specific number of bits is reserved for said at least one parameter, and"

As shown above in *Section IX.A.1.c*, Toombs teaches this claim limitation.

Ex. 1009 at ¶¶175-76.

d. 31(c) "the memory card further configured to have stored therein an addressing data, said addressing data being indicative of at least one addressing method supported,"

As shown above in *Section IX.A.1.d*, Toombs teaches this claim limitation.

Ex. 1009 at ¶¶177-89.

e. 31(d) "wherein the addressing data indicates either a basic addressing method or an expanded addressing method, and"

As shown above in *Section IX.A.1.e*, Toombs teaches this claim limitation.

Ex. 1009 at ¶¶190-95.

f. 31(e) "wherein the memory card is configured so that, if the addressing data indicates that the memory card supports the expanded addressing method, the memory card uses the expanded addressing method in response to a successful reading of the addressing data that indicates support of the expanded addressing method."

As shown above in *Section IX.A.6*, Toombs teaches this claim limitation.

Ex. 1009 at ¶¶221-25.

B. GROUND 2: Claims 6, 8-11, 23, 25-27, and 30-31 are obvious in view of Toombs and Dent

Toombs discloses all elements of Claims 6, 8-11, 25-27 and 30-31. *See* Ground 1. Additionally, those claims, as well as Claim 23, are obvious over Toombs and Dent, as explained below.

Motivation to Combine Toombs and Dent

A POSITA would have been motivated to combine Toombs with the teachings of Dent to achieve a memory card capable of expanded addressing. Ex. 1009 at ¶268. Toombs and Dent are analogous references as they relate to techniques for accessing digital memory and both suggest the desirability of providing a low power memory system that offers high performance while allowing for access to an expanded number of memory locations as card size grows. *Id.* Indeed, both Toombs and Dent describe the respective inventions with reference to applications of memory access in, for example, mobile phones. Ex. 1005 at 1:19-42; Ex. 1006 at 1:46-67.

It would have been obvious to a POSITA to apply Dent's technique of using index register bits to determine an addressing method to the system described in Toombs in order to enable a memory card to operate in a basic or expanded addressing method, depending on the card size. Ex. 1009 at ¶269. Toombs describes that it uses 32-bit addresses to indicate the address of a byte stored in the

memory. Ex. 1005 at Fig. 41 (CMD 24 and CMD17 access a block of the size selected by the SET_BLOCKLEN). In the case of sequential reads, Toombs also discloses the use of the 32-bit address to indicate the first byte of a stream of data. Ex. 1005 at 97. A POSITA would recognize that this addressing method presents a limit as to the size of the address space (in bytes). Ex. 1009 at ¶269.

The express teaching in Dent provides motivation to use its addressing modes to permit greater addressing with fewer bytes, and therefore with lower power, to increase capacity. Ex. 1009 at ¶270; Ex. 1006 at 1:57-67. A POSITA would have been motivated to consider different versions of memory storage and addressing, such as the microprocessor memory system taught in Dent, to enable different addressing methods in a memory card as described in Toombs. Ex. 1009 at ¶270. To this end, a POSITA would be motivated to use a stored bit (or bits) to determine an effective address for a memory reference location, as described in Dent, to the memory card described in Toombs. *Id*.

Further, a POSITA would recognize that the definition or value of a parameter such as C_SIZE could be changed so that it related to the larger blocks of data that an expanded device would address and could continue to be used to calculate the capacity of the expanded memory card. *Id.* at ¶271. Alternatively, if capacity increased and the C_SIZE parameter functioned the same way, a POSITA would consider adding more bits for the parameter C_SIZE to accommodate the

increased address space. *Id.* This combination would also yield a predictable result—a memory card with addressing modes that accommodate greater addressing with fewer bytes, and therefore with lower power, such that capacity of the memory card is increased. *Id.*

1. Independent Claim 6

a. 6 (preamble) "A memory card comprising:"

As shown above in *Section IX.A.1.a*, Toombs teaches this claim limitation. Ex. 1009 at ¶¶168-69.

b. 6(a) "several memory locations for storing data, and in which memory card is stored at least one parameter, the memory card configured so that the number of memory locations of the memory card can be calculated on the basis of said at least one parameter"

As shown above in *Section IX.A.1.b*, Toombs teaches this claim element. Ex. 1009 at ¶¶170-74.

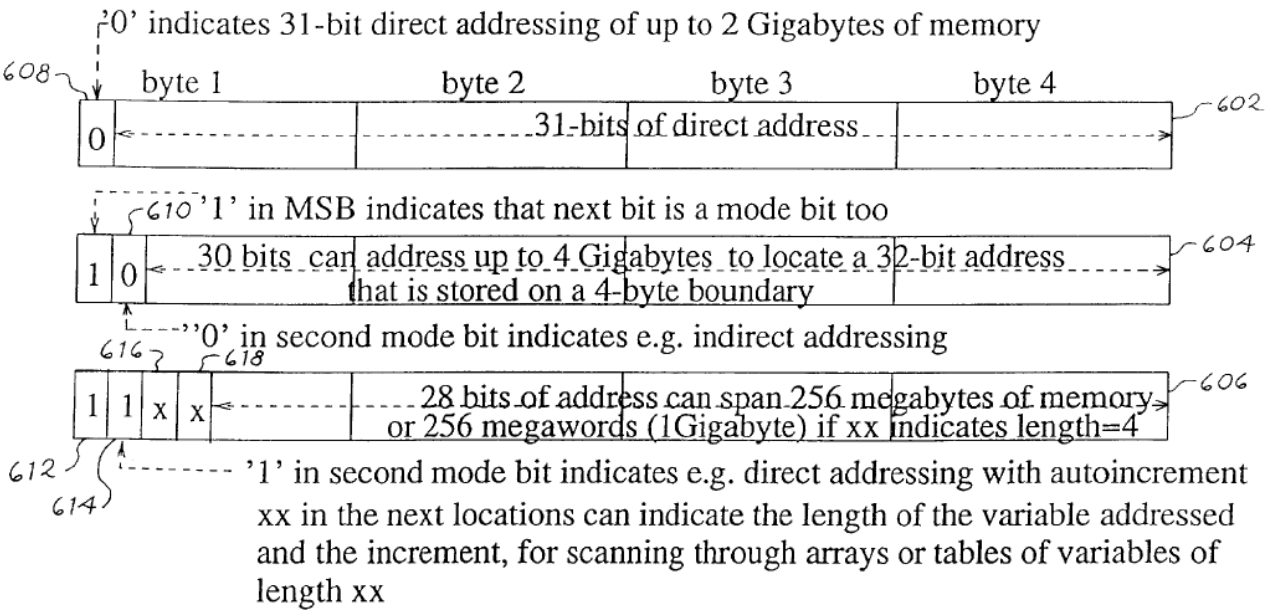
c. 6(b) "the memory card configured so that a specific number of bits is reserved for said at least one parameter, and"

As shown above in *Section IX.A.1.c* Toombs teaches this claim element. Ex. 1009 at ¶¶175-76.

d. 6(c) "the memory card further configured to have stored therein an addressing data, said addressing data being indicative of at least one addressing method supported"

To the extent that the Board determines that Toombs does not disclose this element, it would have been obvious over Toombs in view of Dent. Ex. 1009 at ¶247. Dent discloses a processor architecture and associated method in which the most significant byte of an index register contains mode bits to indicate how the register shall be used in forming the address memory. Ex. 1006 at FIG. 6, 7:1-3. In Dent, index registers are employed in calculating an effective address for a memory-reference instruction. *Id.* at 6:1-3. The index registers are used for reading, manipulating, and storing data to memory. *Id.* at 6:30-34. Each index register comprises a mode byte indicating how the register value shall be used. *Id.* at 6:3-5. Figure 6 of Dent illustrates how the most significant byte of an index register indicates an addressing method:

Fig. 6



Dent describes two modes by which the index registers can use the same number of address bits to be used to access differing amounts of memory. *Id.*; Ex. 1009 at ¶248. Dent explains the desirability of expanding the available range of addresses without the use of additional address bits because the use of fewer bits reduces power consumption. Ex. 1006 at 2:2-8. In one embodiment described in Dent, a microprocessor can use 31 address bits in "byte mode," to address up to 2GB of memory on a single-byte basis. *Id.* at 7:7-10. This first mode is indicated when the value of the most significant bit of the index register is equal to 0. *Id.*

In a second mode described in Dent, the microprocessor can use only 30 of those same address bits to address up to 4GB of memory on a 4-byte basis. *Id.* at 7:22-24. This second mode is indicated when the value of the most significant bit

of the index register is equal to 1, which in turn indicates that the second most significant bit is a mode bit as well. *Id.* at 7:15-17.

Dent therefore discloses this claim limitation. Ex. 1009 at ¶249.

e. 6(d) "wherein the addressing data indicates either a basic addressing method or an expanded addressing method, and"

To the extent that the Board determines that Toombs does not disclose this element, it would have been obvious over Toombs in view of Dent.

Dent offers two modes that permit the same number of address bits to be used to access differing amounts of memory. Ex. 1009 at ¶250. Dent describes a memory system in which a microprocessor can use a fixed range of 31 address bits in one mode, single byte mode, to address up to 2GB of memory. Ex. 1006 at FIG. 6. This mode is a basic addressing method. Ex. 1009 at ¶250. Dent also describes a second mode, 4-byte "word" mode, in which the microprocessor can use only 30 of those same address bits to address up to 4GB of memory. *Id*. This mode is an expanded addressing method. *Id*.

Dent therefore discloses this claim limitation. *Id*.

f. 6(e) "wherein the expanded addressing method enables the addressing of data in a larger number of memory locations than the basic addressing method."

To the extent that the Board determines that Toombs does not disclose this element, it would have been obvious over Toombs in view of Dent.

Dent describes two modes that permit the same number of address bits to be used to access differing amounts of memory. Ex. 1009 at ¶251. The basic addressing method disclosed in Dent enables a microprocessor to address data in 2GB of memory locations. *Id.* The expanded addressing method disclosed in Dent enables a microprocessor to address data in 4GB of memory locations. *Id.* The expanded addressing method therefore enables the addressing of data in a larger number of memory locations than the basic addressing method by enabling addressing of data in 4GB of memory instead of only 2GB of memory. *Id*.

Dent therefore discloses this claim limitation. *Id*.

Because the combination of Toombs and Dent discloses or teaches each and every element of Claim 6, the combination renders obvious Claim 6. Ex. 1009 at ¶252.

2. Dependent Claim 8-11

As shown above in *Sections IX.A.2-IX.A.5*, Toombs describes this claim limitation. Ex. 1009 at ¶¶203-20.

3. Dependent Claim 23 "The memory card according to claim 6, wherein the expanded addressing method supports a higher memory capacity than the basic addressing method."

As discussed in *Section IX.B.1*, Dent offers two modes that permit the same number of address bits to be used to access differing amounts of memory.

Ex. 1009 at ¶¶246-52. The basic addressing method addresses data in 2GB of memory locations. Ex. 1006 FIG. 6; Ex. 1009 ¶254. The expanded addressing method disclosed in Dent enables addressing data in 4GB of memory locations. Ex. 1006 FIG. 6; Ex. 1009 at ¶254. The expanded addressing method described in Dent therefore supports a higher memory capacity than the basic addressing method. Ex. 1009 at ¶254.

Dent therefore discloses this claim. *Id*.

4. Dependent Claim 25 "The memory card according to claim 6, wherein the memory card is configured so that, if the addressing data indicates that the memory card supports the expanded addressing method, the memory card uses the expanded addressing method in response to a successful reading of the addressing data that indicates support of the expanded addressing method."

As discussed in *Section IX.B.1*, Dent discloses an expanded addressing method to address up to 4GB of memory. Ex. 1009 at ¶256. The most significant byte of the index register contains mode bits which indicate how the register is to be interpreted in addressing the memory. *Id*.; Ex. 1006 7:1-3. The expanded addressing method is indicated in the first byte of the 32 bit address stored to an index register. Ex. 1009 at ¶256. A POSITA would understand that the most significant byte of the index register must be successfully read by the microprocessor before an expanded addressing method as described in Dent can be used. *Id.* at ¶257.

Dent therefore discloses this claim. *Id*.

5. Dependent Claim 26 "The memory card according to claim 6, further comprising a register for storing the addressing data."

As discussed in *Section IX.B.1*, Dent describes addressing mode bits that are indicative of an addressing method. Figure 6 of Dent "shows formatting for index registers using addressing mode bits." Ex. 1006 at 2:23-24. Dent therefore discloses a register for storing the addressing data. Ex. 1009 at ¶259.

Dent therefore discloses this claim. *Id.* at 260.

6. Dependent Claim 27 "The memory card according to claim 26, wherein the stored addressing data comprises one bit."

As discussed in *Section IX.B.5*, Dent discloses addressing data stored to a register that indicates a basic addressing method or an expanded addressing method. Ex. 1009 at ¶261. The value 0 in the most significant bit of the index register indicates a basic mode of addressing. Ex. 1006 at FIG. 6; Ex. 1009 at ¶261. Therefore, the single, first bit of the index register constitutes addressing data. Ex. 1009 at ¶261.

It would also be obvious to modify Dent such that a value of 1 in the most significant bit of the index register would identify the expanded mode of addressing. *Id.* at ¶262. In Dent's expanded mode, the system uses the first 2 bits of data to identify the expanded mode (the second bit is used to distinguish

between the expanded mode and a third addressing mode involving an auto-incrementing addressing feature), and uses the remaining 30 of the address bits to address up to 4 GB of memory using 4-byte word addressing. *Id*.; Ex. 1006 7:22-24, FIG. 6.

A POSITA would find it obvious to simplify Dent to use only two addressing modes, distinguished by a single bit in the addressing data stored to the index register. Ex. 1009 at ¶263. This simplification would permit a single bit that indicates either a basic addressing method that permits byte addressing of 2 GB (2^{31} addresses * 1 Byte/address), or an expanded addressing method that permits word addressing allowing increased memory capacity, for example, addressing of 4GB (2^{31} addresses) * (2 Byte/address). *Id.* A POSITA would be motivated to make this simplification, eliminating the third addressing mode of Dent, to permit an additional bit to be used for the address data. *Id.* This would permit a larger number of discrete addresses to be used by the host in the expanded access mode (31 bits versus 30 in Dent), allowing the host more granular access to the stored data. *Id*.

A POSITA, in view of Dent's teachings, would recognize the benefit of this approach, which maintains the same number of total bits (32 in the case of Dent) used to convey addressing information, while permitting the host to address larger total memory capacities. Ex. 1009 at ¶264. This would avoid the need to redesign

the interface addressing interface between the host and the memory device, while still permitting the host to take advantage or larger capacity memory devices. *Id.* Moreover, a POSITA would recognize this approach would allow the system designer to easily expand this system to access larger memory arrays by simply enlarging the size of the word (i.e., the number of bytes in memory accessed with a single address). *Id*.

Furthermore, this claim element uses the open term "comprises" rather than a closed term, such as "consisting of." The term "comprises one bit" means "at least one bit," so if there are more bits stored to a register, this limitation is still satisfied. *Id.* at ¶265; *see, e.g., Mars Inc. v. H.J. Heinz Co*., 377 F.3d 1369, 1376 (Fed. Cir. 2004) (the term "comprising" is inclusive or open-ended and does not exclude additional, unrecited elements or method steps).

7. Independent Claims 30-31

As shown above in *Sections IX.B.1* and *IX.B.4,* Toombs in view of Dent render Claims 30-31 obvious. Ex. 1009 at ¶¶246-52, 256-58.

C. GROUND 3: Claims 6, 8-11, 23, 25-27, and 30-31 are obvious in view of Toombs and PCMCIA

Toombs discloses all elements of Claims 6, 8-11, 25-27 and 30-31. *See* Ground 1. Additionally, those claims, as well as Claim 23, are obvious over Toombs and PCMCIA, as explained below.

Motivation to Combine Toombs and PCMCIA

A POSITA would have been motivated to combine Toombs with the teachings of PCMCIA to achieve a memory card capable of expanded addressing. Ex. 1009 at ¶296. A POSITA would have recognized that the memory card described in Toombs could not address more than 2GB of data, and would have been motivated to consider how the other leading mass storage standards had solved this problem when they reached the limits of the addressing methods inherent in their design. *Id.* The prior art in mass storage devices, such as removable flash memory cards in the case of PCMCIA, encountered the same problem as Toombs of limited capacity. *Id.* Therefore, there was a known need for the capability to access a larger number of addresses than the original standards allowed. *Id*.

In PCMCIA, the standard was expanded by increasing the number of address bits so that a larger number of addresses could be accessed. *Id.* at ¶297. In order for a host to know whether a particular PCMCIA card used the expanded addressing method, or only worked with limited addressing of the original standard, a tuple was stored in the device so that the host could read that data and, based on that data, the host could know that it should use the expanded addressing method. *Id*.

It would be obvious to a POSITA to store the information indicative of the addressing mode, i.e., the information indicating that a higher number of address bits are available, in a register, such as the CSD register described in Toombs. *Id.* at ¶298. The CIS (Card Information Structure) described in PCMCIA functions in the same manner as the CSD register of Toombs—it stores information about the functions and data format of the card so that the host can read the information and know the constraints of the memory card. *Id.* Accordingly, a POSITA would have been motivated to substitute a register for a CIS to achieve predictable results—easy access by the host to information about the card's function and format. *Id*.

Further, a POSITA would understand that modifying Toombs with the teaching of PCMCIA would enable a memory system to be able to calculate the capacity of the card based on a stored parameter such as C_SIZE. *Id.* at ¶299. It would have been obvious to a POSITA to continue to use a parameter such as C_SIZE, and to expand the number of bits of the parameter to accommodate a larger capacity. *Id.*

1. Independent Claim 6

a. 6 (preamble) "A memory card comprising:"

As shown above in *Section IX.A.1.a*, Toombs teaches this claim limitation. Ex. 1009 at ¶¶168-69.

b. 6(a) "several memory locations for storing data, and in which memory card is stored at least one parameter, the memory card configured so that the number of memory locations of the memory card can be calculated on the basis of said at least one parameter"

As shown above in *Section IX.A.1.b*, Toombs teaches this claim element. Ex. 1009 at ¶¶170-74.

c. 6(b) "the memory card configured so that a specific number of bits is reserved for said at least one parameter, and"

As shown above in *Section IX.A.1.c* Toombs teaches this claim element. Ex. 1009 at ¶¶175-76.

d. 6(c) "the memory card further configured to have stored therein an addressing data, said addressing data being indicative of at least one addressing method supported"

To the extent that the Board determines that Toombs does not disclose this element, the element would be obvious over Toombs in view of PCMCIA. Ex. 1009 at ¶¶272-73.

PCMCIA defines specifications for PC Cards and communications to and from hosts. Ex. 1007 at 1. PCMCIA describes multiple ways of expanding addressing for a PC card memory system. Ex. 1009 at ¶¶274-77. One way is the use of additional address bits. *Id*.

PCMCIA describes a Configuration Option register which is used to configure the card and provide an address extension to select a 64 MB page of Common Memory. *Id.* at ¶276; Ex. 1007 at 75. The Configuration Option register's Common Memory Address Extension field provides six address extension bits, as shown in Table 4-29 below. Ex. 1007 at 76, Table 4-29.

Table 4–29 Configuration Option Register

D7	D6	D5	D4	D3	D2	D1	D0
SRESET	LevlREQ	Function Configuration Index / Common Memory Address Extension					

Common Memory Address Extension	R/W	For a memory card with > 64 MBytes of Common Memory and using 64 MByte paging this field can be used to provide six address extension bits which select a 64 MByte page within a Common Memory space as large as 4 Gigabytes. The selection of the Common Memory Address Extension field option for use with 64 MByte paging as well as the exact size of Common Memory is specified by the CISTPL_EXTDEVICE tuple.

The selection of the Common Memory Address Extension field option for use with 64MByte paging as well as the exact size of Common Memory is specified by the CISTPL_EXTDEVICE tuple. *Id*. This tuple is contained in the memory card's Card Information Structure (CIS), which is stored in the attribute memory of the card. *Id.* at 21, 35; Ex. 1009 at ¶276. This tuple is therefore addressing data indicative of at least one supported addressing method. Ex. 1009 at ¶¶274-277.

e. 6(d) "wherein the addressing data indicates either a basic addressing method or an expanded addressing method, and"

To the extent that the Board determines that Toombs does not disclose this element, the element would be obvious over Toombs in view of PCMCIA. Ex. 1009 at ¶278.

PCMCIA discloses addressing indicative of at least one addressing method. *Id.* PCMCIA discloses a basic addressing method, wherein the card neither contains an Address Extension Register nor uses the Configuration Option Register for address extension. Ex. 1007 at 34, Table 4-3; Ex. 1009 at ¶278. PCMCIA also discloses an expanded addressing method, wherein the card uses the Configuration Option Register to employ an additional six address bits for address extension. *Id*.

f. 6(e) "wherein the expanded addressing method enables the addressing of data in a larger number of memory locations than the basic addressing method."

PCMCIA discloses addressing data indicative of either an expanded or a basic addressing method. *Id*.

The basic addressing method disclosed in PCMCIA enables addressing of 64MB of memory. Ex. 1007 at 34, Table 4-3; Ex. 1009 at ¶279. The expanded addressing method disclosed in PCMCIA enables addressing of 4GB of memory. Ex. 1007 at 34, Table 4-3, 14; Ex. 1009 at ¶300. PCMCIA therefore discloses

enabling the addressing of data in a larger number of memory locations than the basic addressing method. Ex. 1009 at ¶279.

2. Dependent Claim 8-11

As shown above in *Sections IX.A.2-IX.A.5*, Toombs describes this claim limitation. Ex. 1009 at ¶¶203-20.

3. Dependent Claim 23 "The memory card according to claim 6, wherein the expanded addressing method supports a higher memory capacity than the basic addressing method."

PCMCIA discloses addressing data indicative of either an expanded or a basic addressing method. Ex. 1009 at ¶282.

The basic addressing method disclosed in PCMCIA enables addressing of 64MB of memory, while the expanded method allows addressing 4 GB, through 6 additional address bits. Ex. 1007 at Table 4-3, 14, 56; Ex. 1009 at ¶282. PCMCIA therefore discloses an expanded addressing method that supports a higher memory capacity than the basic addressing method. Ex. 1009 at ¶¶282-83.

4. Dependent Claim 25 "The memory card according to claim 6, wherein the memory card is configured so that, if the addressing data indicates that the memory card supports the expanded addressing method, the memory card uses the expanded addressing method in response to a successful reading of the addressing data that indicates support of the expanded addressing method."

PCMCIA discloses an expanded addressing method wherein the Common Memory Address Extension field option is used with 64MB paging. Ex. 1009 at

¶285. Use of the Common Memory Address Extension field is specified by the CISTPL_EXTDEVICE tuple. Ex. 1007 at 76, Table 4-29. This tuple is stored in the memory card's Card Information Structure (CIS) in the attribute memory of the card. *Id.* at 21, 35. While in the power-up state, the PC card shall allow the CIS to be read and Configuration registers to be accessed. *Id.* at 74. A POSITA would understand that the CISTPL_EXTDEVICE must be successfully read by the microprocessor before an expanded addressing method as described in PCMCIA can be used. Ex. 1009 at ¶¶285-86.

5. Dependent Claim 26 "The memory card according to claim 6, further comprising a register for storing the addressing data."

The addressing method is specified by the CISTPL_EXTDEVICE tuple. Ex. 1007 at 76, Table 4-29; Ex. 1009 ¶288. The CISTPL_EXTDEVICE tuple is stored in the Card Information Structure (CIS) in attribute memory. Ex. 1007 at 21, 35, 187. A POSITA would understand that the CIS is a register. Ex. 1009 at ¶288. The CIS of a PC Card stores tuples that identify functions and data formats for the card. *Id.*; Ex. 1007 at 21. Accordingly, the CIS stores the addressing data of the CISTPL_EXTDEVICE that indicates whether the card supports a basic or an expanded addressing method. Ex. 1009 at ¶288.

To the extent that the CIS of the PC card is not a register, it would be obvious to modify the PC Card to store information about the functions and data

format of the card in a register, such as the CSD register of the MMC card described in Toombs. Ex. 1009 at ¶289. It would obvious to a POSITA to modify the memory card of PCMCIA to store information indicative of an addressing method in a register. *Id*. at ¶¶289, 296-99.

6. Dependent Claim 27 "The memory card according to claim 26, wherein the stored addressing data comprises one bit."

PCMCIA discloses a tuple stored in the Card Information Structure (CIS) in attribute memory that indicates an expanded or basic addressing method. Ex. 1009 at ¶291. It would be obvious to modify PCMCIA to store the addressing data as a single bit in the CIS in order to indicate whether the card supports expanded addressing to minimize the size of the register stack and increase processing speed. *Id*.

To the extent that the CIS of the PC card is not a register, it would be obvious to modify the PC Card to store information about the functions and data format of the card in a register, such as the CSD register of the MMC card described in Toombs. *Id.* at ¶292. For at least the reasons described in *Section IX.C*, it would obvious to a POSITA to modify the memory card of PCMCIA to store information indicative of an addressing method in a register. *Id.*

Furthermore, this claim element uses the term "comprises one bit." "Comprises one bit" means "includes at least one bit," so if there is more than one

bit stored to a register, this limitation is still satisfied. *See, e.g., Mars Inc. v. H.J. Heinz Co*., 377 F.3d at 1376. PCMCIA describes a tuple of at least one bit stored to the memory card as addressing data indicative of an addressing method. Ex. 1007 at 76; Ex. 1009 at ¶292-93.

7. Independent Claims 30-31

As shown above in *Sections IX.C.1* and *IX.C.4,* Toombs in view of PCMCIA renders Claims 30 and 31 obvious. *See also* Ex. 1009 at ¶¶272-80, 285-87.

D. GROUND 4: Claims 6, 8-11, 23, 25-27, and 30-31 are obvious in view of Toombs and Revision 3a of ATA-6

Toombs discloses all elements of Claims 6, 8-11, 25-27 and 30-31. *See* Ground 1. Additionally, those claims, as well as Claim 23, are obvious over Toombs and Revision 3a of the ATA-6 Standard, as explained below.

Motivation to Combine Toombs and Revision 3a of the ATA-6 Standard

A POSITA would have been motivated to combine Toombs with the teachings of Revision 3a of the ATA-6 Standard to achieve a memory card capable of expanded addressing. Ex. 1009 at ¶320. A POSITA would have recognized that the memory card described in Toombs could not address more than 2GB of data, and would have been motivated to consider how the other leading mass storage standards had solved this problem when they reached the limits of the addressing methods inherent in their design. *Id.* The prior art in mass storage

devices, such as disk drives in the case of Revision 3a of the ATA-6 Standard, encountered the same problem as Toombs of limited capacity. *Id.* Therefore, there was a known need for the capability to access a larger number of addresses than the original standards allowed. *Id*.

In Revision 3a of the ATA-6 Standard, the standard was expanded by increasing the number of address bits so that a larger number of addresses could be accessed. *Id.* at ¶321. For a host to know whether a particular card used the expanded addressing method, or only worked with limited addressing of the original standard, a bit was stored in the device so that the host could read that bit and, based on that bit, the host could know that it should use the expanded addressing method. *Id*.

Further, a POSITA would understand that modifying Toombs with the teaching of Revision 3a of the ATA-6 Standard would enable a memory system to be able to calculate the capacity of the card based on a stored parameter such as C_SIZE. *Id.* at ¶322. It would have been obvious to a POSITA to continue to use a parameter such as C_SIZE, and to expand the number of bits of the parameter to accommodate a larger capacity. *Id.*

1. Independent Claim 6

a. 6 (preamble) "A memory card comprising:"

As shown above in *Section IX.A.1.a*, Toombs teaches this claim limitation. Ex. 1009 at ¶¶168-69.

b. 6(a) "several memory locations for storing data, and in which memory card is stored at least one parameter, the memory card configured so that the number of memory locations of the memory card can be calculated on the basis of said at least one parameter"

As shown above in *Section IX.A.1.b*, Toombs teaches this claim element. Ex. 1009 at ¶¶170-74.

c. 6(b) "the memory card configured so that a specific number of bits is reserved for said at least one parameter, and"

As shown above in *Section IX.A.1.c*, Toombs teaches this claim element. Ex. 1009 at ¶¶175-76.

d. 6(c) "the memory card further configured to have stored therein an addressing data, said addressing data being indicative of at least one addressing method supported"

To the extent that the Board determines that Toombs does not disclose this element, it would have been obvious over Toombs in view of Revision 3a of the ATA-6 Standard. Ex. 1009 at ¶301.

Revision 3a of the ATA-6 Standard specifies the AT Attachment Interface between host systems and storage devices. Ex. 1008 at 1. This interface is used by CompactFlash storage devices. *Id.* at 3. Revision 3a of the ATA-6 Standard describes a data register used for sending commands to the device or posting status from the device. *Id.* at 63. The IDENTIFY DEVICE command enables the host to receive parameter information from the device. *Id.* at 114. This command allows the host to read device identification data from Data register. *Id.* Table 27 of Revision 3a of the ATA-6 Standard defines the arrangement and meaning of the parameter words in the buffer, shown below. *Id.* at 117.

Table 27 – IDENTIFY DEVICE information *(continued)*

Word	O/M	F/V		Description
81	M	F		Minor version number 0000h or FFFFh = device does not report version 0001h-FFFEh = see 8.15.41
82	M			Command set supported.
		X	15	Obsolete
		F	14	1 = NOP command supported
		F	13	1 = READ BUFFER command supported
		F	12	1 = WRITE BUFFER command supported
		X	11	Obsolete
		F	10	1 = Host Protected Area feature set supported
		F	9	1 = DEVICE RESET command supported
		F	8	1 = SERVICE interrupt supported
		F	7	1 = release interrupt supported
		F	6	1 = look-ahead supported
		F	5	1 = write cache supported
		F	4	Shall be cleared to zero to indicate that the PACKET Command feature set is not supported.
		F	3	1 = mandatory Power Management feature set supported
		F	2	1 = Removable Media feature set supported
		F	1	1 = Security Mode feature set supported
		F	0	1 = SMART feature set supported
83	M			Command sets supported.
		F	15	Shall be cleared to zero
		F	14	Shall be set to one
		F	13	1 = FLUSH CACHE EXT command supported
		F	12	1 = mandatory FLUSH CACHE command supported
		F	11	1 = Device Configuration Overlay feature set supported
		F	10	1 = 48-bit Address feature set supported
		F	9	1 = Automatic Acoustic Management feature set supported
		F	8	1 = SET MAX security extension supported
		F	7	See Address Offset Reserved Area Boot, NCITS TR27:2001
		F	6	1 = SET FEATURES subcommand required to spinup after power-up
		F	5	1 = Power-Up In Standby feature set supported
		F	4	1 = Removable Media Status Notification feature set supported
		F	3	1 = Advanced Power Management feature set supported
		F	2	1 = CFA feature set supported
		F	1	1 = READ/WRITE DMA QUEUED supported
		F	0	1 = DOWNLOAD MICROCODE command supported

Revision 3a of the ATA-6 Standard describes that bit 10 of word 83 of the device identification data indicates whether 48-bit addressing is supported. *Id.* at 36-37, 131. This bit is addressing data indicative of at least one addressing method. Ex. 1009 at ¶¶301-03. If this field is not enabled, 28-bit addressing is supported. *Id.* at ¶303.

e. 6(d) "wherein the addressing data indicates either a basic addressing method or an expanded addressing method, and"

To the extent that the Board determines that Toombs does not disclose this element, it would have been obvious over Toombs in view of Revision 3a of the ATA-6 Standard. Ex. 1009 at ¶304.

Revision 3a of the ATA-6 Standard describes addressing data indicative of at least one addressing method. Ex. 1009 at ¶305. The optional 48-bit address feature set allows devices with capabilities up to 281 tera sectors. Ex. 1008 at 51. A sector is a uniquely addressable set of 512 bytes. *Id.* at 5. This allows device capacity of up to approximately 144 petabytes. *Id.* Accordingly, the addressing data, bit 10 of word 38 in the Data register, indicates that either the expanded addressing method (48-bit addressing) is enabled or that the basic addressing method (28-bit addressing) is enabled. Ex. 1009 at ¶¶304-05.

f. 6(e) "wherein the expanded addressing method enables the addressing of data in a larger number of memory locations than the basic addressing method."

To the extent that the Board determines that Toombs does not disclose this element, it would have been obvious over Toombs in view of Revision 3a of the ATA-6 Standard. *Id.* at ¶306.

Revision 3a of the ATA-6 Standard describes a bit in a data register that indicates either an expanded or a basic addressing method. *Id.* at ¶307. The

expanded addressing method described in Revision 3a of the ATA-6 Standard allows a device to address approximately 144 petabytes. Ex. 1008 at 51. Where 48-bit addressing is not enabled, the device operates in a basic addressing method using 28-bit addresses. *Id.* In this basic addressing method, the device capacity is limited to 2^{28} (268,435,456) sectors of 512 bytes each for a total of 137 gigabytes. *Id.*; Ex. 1009 at ¶315. The expanded addressing method described in Revision 3a of the ATA-6 Standard therefore enables the address of data in a larger number of memory locations than the basic addressing method. Ex. 1009 at ¶¶306-07.

2. Dependent Claim 8-11

As shown above in *Sections IX.A.2-IX.A.5*, Toombs discloses these claims. Ex. 1009 at ¶¶203-20.

3. Dependent Claim 23 "The memory card according to claim 6, wherein the expanded addressing method supports a higher memory capacity than the basic addressing method."

Revision 3a of the ATA-6 Standard describes a bit in a data register that indicates either an expanded or a basic addressing method. Ex. 1009 at ¶310. The expanded addressing method described in Revision 3a of the ATA-6 Standard allows a device to address approximately 144 petabytes. Ex. 1008 at 65. Where 48-bit addressing is not enabled, the device operates in a basic addressing method using 28-bit addresses. *Id.* In this basic addressing method, the device capacity is

limited to 2^{28} (268,435,456) sectors of 512 bytes each for a total of 137 gigabytes. *Id.* The expanded addressing method described in ATA-6 therefore supports a higher memory capacity than the basic addressing method. Ex. 1009 at ¶¶311-12.

4. Dependent Claim 25 "The memory card according to claim 6, wherein the memory card is configured so that, if the addressing data indicates that the memory card supports the expanded addressing method, the memory card uses the expanded addressing method in response to a successful reading of the addressing data that indicates support of the expanded addressing method."

Revision 3a of the ATA-6 Standard describes a data register that stores addressing data indicative of an expanded or basic addressing method. Ex. 1009 at ¶312. The IDENTIFY DEVICE command enables the host to receive parameter information from the device. Ex. 1008 at 128. This command allows the host to read the 256 words of device identification data from Data register, including bit 10 of word 38 that indicates an addressing method. *Id*.; Ex. 1009 at ¶312.

A POSITA would understand that the device identification information in the Data Register must be read by the host before an expanded addressing method as described in Revision 3a of the ATA-6 Standard can be used. Ex. 1009 at ¶313.

5. Dependent Claim 26 "The memory card according to claim 6, further comprising a register for storing the addressing data."

Revision 3a of the ATA-6 Standard describes a data register that stores addressing data indicative of an expanded or basic addressing method. Ex. 1008 at

77. Bit 10 of word 83, addressing data, is stored in the data register to be read by the host. Ex. 1008 at 36-37, 131; Ex. 1009 at ¶¶315-16. Revision 3a of the ATA-6 Standard therefore describes a register for storing the addressing data. Ex. 1009 at ¶¶315-16.

6. Dependent Claim 27 "The memory card according to claim 26, wherein the stored addressing data comprises one bit."

Revision 3a of the ATA-6 Standard describes a data register that stores addressing data indicative of an expanded or basic addressing method. Ex. 1009 at ¶317. Bit 10 of word 83, addressing data, is stored in the data register to be read by the host. *Id*.; Ex. 1008 at 36, 131. Revision 3a of the ATA-6 Standard therefore describes addressing data comprising one bit stored to a register. Ex. 1009 at ¶317.

7. Independent Claims 30-31

As shown above in *Sections IX.D.1* and *IX.D.4,* Toombs in view of Revision 3a of the ATA-6 Standard renders Claims 30 and 31 obvious. *See also* Ex. 1009 at ¶¶300-308, 312-14.

X. CONCLUSION

For the foregoing reasons, Petitioner respectfully requests that a trial for *inter partes* review of the RE486 Patent be instituted and that Claims 6, 8-11, 23, 25-27, and 30-31 be rejected and canceled.

Dated: January 30, 2019

Respectfully submitted,

/Robert C.F. Pérez/

Robert C.F. Pérez (Reg. No. 39,328)
Christopher Kao (*Pro hac vice* to be filed)
Brock S. Weber (*Pro hac vice* to be filed)

Attorneys for Petitioner
Kingston Technology Company, Inc.

PILLSBURY WINTHROP SHAW PITTMAN LLP
1650 Tysons Boulevard, 14th Floor
McLean, VA 22102
Telephone: 703.770.7900
Facsimile: 703.770.7901

CERTIFICATE OF COMPLIANCE

The undersigned certifies that this brief complies with the type volume limitations of 37 CFR § 42.24(a)(1)(i). This brief contains 13,642 words (excluding the table of contents, the table of authorities, mandatory notices under 37 CFR § 42.8, the certificate of service, certificate of compliance, and appendix of exhibits), as calculated by the "Word Count" feature of Microsoft Word Document, the word processing program used to create it.

The undersigned further certifies that this brief complies with the typeface requirements of 37 CFR § 42.6(a)(2)(ii) and typestyle requirements of 37 CFR § 42.6(a)(2)(iii). This brief has been prepared in a proportionally spaced typeface using Microsoft Word Document in Times New Roman 14 point font.

Dated: January 30, 2019

Respectfully submitted,

/Robert C.F. Pérez/

PILLSBURY WINTHROP SHAW PITTMAN LLP
1650 Tysons Boulevard, 14th Floor
McLean, VA 22102
Telephone: 703.770.7900
Facsimile: 703.770.7901

Robert C.F. Pérez (Reg. No. 39,328)
Christopher Kao (*Pro hac vice* to be filed)
Brock S. Weber (*Pro hac vice* to be filed)

Attorneys for Petitioner
Kingston Technology Company, Inc.

CERTIFICATE OF SERVICE

The undersigned hereby certifies that a true copy of the foregoing

PETITION FOR *INTER PARTES* REVIEW OF U.S. PATENT NO. RE45,486

and supporting materials (Exhibits 1001 - 1015 and Power of Attorney) have been served in its entirety this January 30, 2019, by e-mail and Federal Express on Patent Owner at the correspondence address for the attorney of record for the RE45,486 Patent shown in USPTO PAIR, as well as on counsel for Patent Owner in the co-pending litigation:

Patent Owner in PAIR (*via FedEx*):

Jason Eisenberg
Sterne, Kessler, Goldstein & Fox P.L.L.C.
1100 New York Avenue, N.W.
Washington, DC 20005

Counsel for Patent Owner in co-pending litigation (*via electronic mail*):

Andrew G. Strickland
Andrew.Strickland@leehayes.com
William B. Dyer III
Bill.Dyer@leehayes.com
LEE & HAYES, P.C.
1175 Peachtree Street
100 Colony Square, Suite 2000
Atlanta, GA 30361

Marc E. Hankin
Marc@HankinPatentLaw.com
Anooj Patel
Anooj@HankinPatentLaw.com
HANKIN PATENT LAW, APC
4299 MacArthur Boulevard,
Suite 100
Newport Beach, CA 92660

/Robert C.F. Pérez/
Robert C.F. Pérez (Reg. No. 39,328)