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. 7,565,469

PETITION FOR *INTER PARTES* REVIEW OF U.S. PATENT NO. 7,565,469

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

Cases

Statutes and Codes

Rules and Regulations

Other Authorities

EXHIBIT LIST

Exhibit	Description
1001	U.S. Patent No. 7,565,469 to Mylly et al. ("the '469 Patent")
1002	File History of the '469 Patent
1003	The MultiMediaCard System Specification, Version 3.31 ("MMC 3.31")
1004	CompactFlash Specification Revision 1.3 ("CompactFlash")
1005	Affidavit of Christopher Butler, Internet Archive
1006	Declaration of Michael Asao
1007	Hitachi, Ltd., MultiMediaCard Product Manual, ADE-603-002 (Rev. 1.0 2000)
1008	Arthur L. Dexter, Microcomputer Bus Structures and Bus Interface Design (1986)
1009	SanDisk Corp., CompactFlash Memory Card Product Manual, Lit. No. 20-10-00038 (Rev. 10.0 2002)
1010	U.S. Patent No. 6,901,457 to Toombs et al.
1011	U.S. Patent No. 6,426,893 to Conley et. al.
1012	U.S. Patent No. 5,811,997 to Chengson et. al.
1013	U.S. Patent No. 4,437,022 to Meirsch et. al.
1014	U.S. Patent No. 5,781,050 to Russell
1015	Declaration of Dr. R. Jacob Baker
1016	Tanenbaum, A., Structured Computer Organization (1990)

Exhibit	Description
1017	U.S. Provisional Application No. 60/629,098 ("the '098 Application")
1018	Third Joint Claim Construction and Prehearing Statement (N.D. Cal. Patent L.R. 4-3), filed in the related matter on Nov. 16, 2018

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 19 and 20 of U.S. Patent No. 7,565,469 (the "'469 Patent") to Mylly (Ex. 1001), and cancel those claims as unpatentable. The prior art presented in this Petition are The MultiMediaCard System Specification, Version 3.31 (Ex. 1003, "MMC 3.31") and CompactFlash Specification Revision 1.3 (Ex. 1004, "CompactFlash").

As discussed in detail below, MMC 3.31 renders obvious Claims 19 and 20 under 35 U.S.C. § 103, and the combination of MMC 3.31 and CompactFlash render obvious Claims 19 and 20 under 35 U.S.C. § 103. Thus, there is a reasonable likelihood that Petitioner will prevail with respect to the challenged claims, and Petitioner respectfully requests that the Board institute a trial for *inter partes* review and cancel Claims 19 and 20 as unpatentable.

II. MANDATORY NOTICES

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 Claims 19 and 20 of the '469 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 will be filing petitions for *inter partes* review of the other seven patents that MTL has asserted against it.

C. Counsel and Service Information (37 C.F.R. §§42.8(b)(3) & (b)(4))

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-469ipr@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

Pursuant to 37 C.F.R. §42.104(b), Petitioner requests that Claims 19 and 20 of the '469 Patent be cancelled as rendered obvious by MMC 3.31 (Ex. 1003) under 35 U.S.C. § 103, and that Claims 19 and 20 of the '469 Patent be cancelled as rendered obvious by the combination of MMC 3.31 (Ex. 1003) and CompactFlash (Ex. 1004) under 35 U.S.C. § 103.

The '469 Patent was filed on October 14, 2005 and claims priority to a US provisional application no. 60/629,098 ("the '098 Application") filed on

November 17, 2004. Ex. 1001; Ex. 1017. Thus, the earliest potential priority date is November 17, 2004.[1]

The Declaration of R. Jacob Baker, Ph.D., P.E., filed herewith (Ex. 1015), supports the challenge in this Petition that Claims 19 and 20 of the '469 Patent are invalid.

IV. BACKGROUND OF THE TECHNOLOGY

The '469 Patent is generally directed to communication between memory cards and hosts, and in particular, the signaling of the status of the memory card to the host. Ex. 1015 at ¶74. In this field, data is typically transferred from the host to the memory card in the form of blocks of data, each of which consists of multiple bytes. Ex. 1001 at 1:32-42; Ex. 1003 at 26 ("The width of the data path should be a byte; the units which are handling data should work on bytes or blocks of bytes. This requirement is derived from the MMC bus protocol, which is organized in data blocks. Blocks are multiples of bytes."). The block size for memory cards is typically much smaller than the block size of host devices. Ex. 1001 at 1:32-39. For example, host devices often transfer blocks of 16 kilobytes, while cards that support the MMC specification typically support blocks of 512 bytes. *Id.* at 1:32-42. It is thus often advantageous to use multiple block

[1] Petitioner does not concede entitlement to this priority date.

write commands, which obviate the need for a new write block command between each block of data. *Id.* at 1:36-39. Accordingly, many technical standards for memory cards, including the MultiMediaCard System Specification, Version 3.31 (Ex. 1003, "MMC 3.31") and the CompactFlash Specification Revision 1.3 (Ex. 1004, "CompactFlash"), include multiple block write commands. Ex. 1003 at 38; Ex. 1004 at 80.

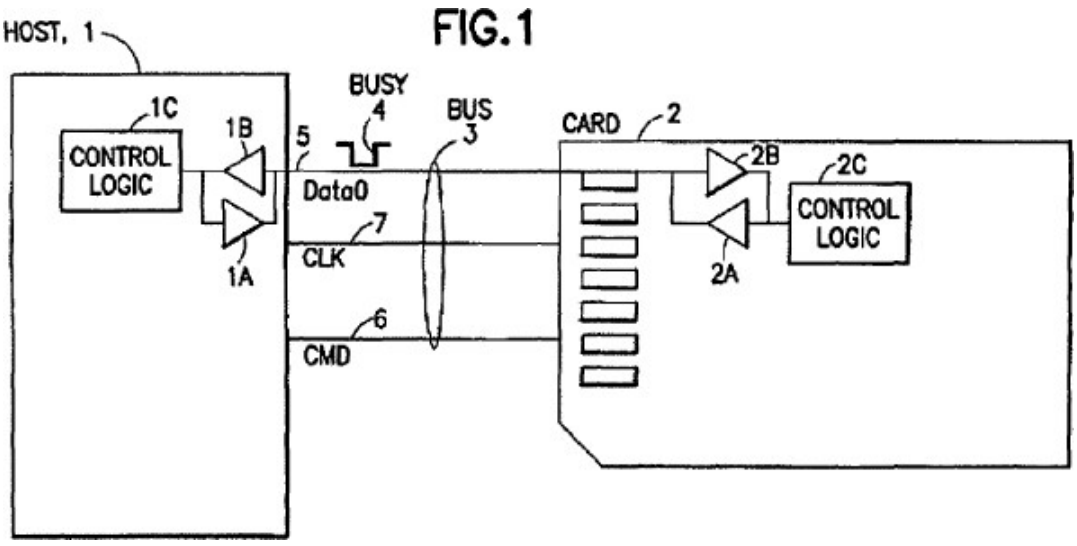
A common issue that must be addressed for multiple block write commands is how a memory card communicates its status to a host device during a data transfer. Ex. 1015 at ¶86. One way to address this issue is to require the host to poll the memory card's status periodically. *Id.* However, other methods were also common. For example, CompactFlash supports the Write Sector(s) command, wherein a host writes a pre-defined number of sectors of data to the CompactFlash card. Ex. 1004 at 80. The card indicates its status to the host using an interrupt signal. *Id*. After each sector, except the last one, the interrupts tell the host that a card buffer is free to receive additional data. *Id*. After the card receives the last data sector, the interrupt signal indicates when the command has been completed (i.e., the data has been programmed). *Id.* The use of this interrupt signaling eliminates the need for the host device to poll the CompactFlash card to determine its status. Ex. 1015 at ¶182.

MMC 3.31 employs a similar signaling technique. MMC 3.31 discloses a multiple block write command, which writes multiple data blocks to the MMC card. Ex. 1003 at 38. The MMC card asserts a busy signal on its data line to signal its status to the host. Ex. 1003 at 22, Fig. 10. For all blocks except the last block, the busy signal indicates that the MMC card's buffers are full. Ex. 1003 at 35.

MMC 3.31 provides for a multiple block write operation that requires that the host send a stop command to the MMC card to terminate the data transfer. Ex. 1003 at 38, Figs. 10, 30-34. The MMC card then sets the same busy signal previously used for buffer status to indicate that the card is programming. Ex. 1003 at Fig. 34. MMC 3.31 also provides for a multiple block write operation that allows the host to set the number of blocks to be transferred beforehand and does not require a stop command. Ex. 1003 at 38.

V. OVERVIEW OF THE '469 PATENT

The '469 Patent purports to disclose a way of signaling the card status to indicate that a buffer is busy/ready or that the card is programming in conjunction with a multiple block write command. Ex. 1001 at Abstract, 4:4-15; Ex. 1015 at ¶93. Figure 1 of the '469 Patent illustrates an exemplary host and memory card connected over a bus:



In this embodiment, the memory card and host communicate via three lines: data line 5, command (CMD) line 6, and clock (CLK) line 7. Ex. 1001 at 3:47-51. Data line 5 (Data0) is used both to transmit data and to assert a busy signal to tell the host when the memory card is able to accept new data. *Id.* 3:47-49, 4:42-44, 4:64-5:2, Fig. 2. The '469 Patent states that "the bus 3 may be compatible with one defined as 'The MultiMediaCard, System Specification, Version 3.31, MMCA Technical Committee', 2003," (MMC) except as modified in accordance with the invention of the '469 Patent. *Id.* at 3:52-55; Ex. 1015 at ¶94.

The '469 Patent is directed to a purported problem with existing methods of signaling that required the host device in an MMC system to poll the programming

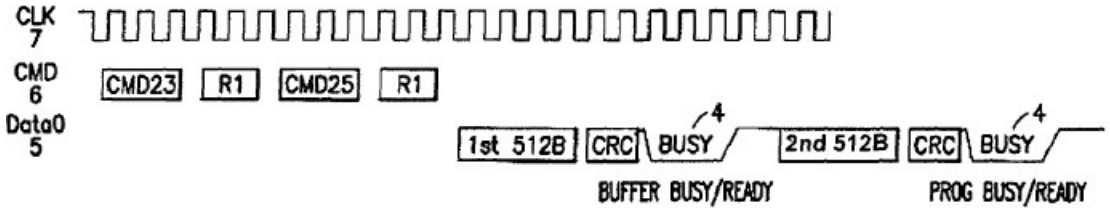
ready status signal to determine the programming state of the card.[2] Ex. 1001 at 1:43-50. In response to such polling, the memory card communicates its programming status. *Id.* According to the '469 Patent, such polling was "an inefficient use of the host's processing capacity." *Id.*

The '469 Patent overcomes this purported inefficiency by asserting the busy signal on the Data0 line after the last data block is received by the memory card to indicate programming status. *Id.* at 4:4-15, 4:43-52. Rather than indicate "buffer busy/ready," as the signal does before each data block is received from the host, the same busy signal, after the last block arrives, indicates "programming busy/ready." *Id.* at 4:6-10, 4:44-52. "This means that the host 1 does not need to begin polling the programming status of the MMC 2, but can instead continue to wait for a busy interrupt in this phase of the access as well. However, the occurrence of the busy interrupt is interpreted by the host 1 as an occurrence of the 'programming ready' status indication." *Id.* at 4:52-57.

[2] As discussed below, the '469 Patent is incorrect, as this is inaccurate for multiple block write commands.

As illustrated in Figure 2 of the '469 Patent, this claimed solution uses a command (CMD23) in addition to the multiple block write command (CMD25) that informs the memory card how many blocks are to be sent from the host so that the card knows when the block transfer is complete. Ex. 1001 at 5:14-20, Fig. 2. This additional command (CMD23) allows the memory card to determine when the last block has been received and to change the meaning of the signal sent to the host from "buffer busy/ready" to "programming busy/ready" (even though the same signal is sent to the host). Ex. 1001 at 4:44-52. More specifically, using CMD23, which is also described in MMC 3.31, the host tells the device the number of blocks of data to be transferred by the next multiple block write command (CMD25), which is also described in MMC 3.31. Ex. 1001 at 5:14-16, Fig. 2; Ex. 1015 at ¶97.

FIG.2



(Ex. 1001 Fig. 2).

The first 512 byte data block is sent from the host to the memory card on the Data0 pin. *Id.* at 5:17-18, Fig. 2. After receiving the first data block, the memory card asserts busy signal 4, which indicates "buffer busy/ready." *Id.* at 5:18-19,

Fig. 2. After the memory card de-asserts the busy signal to indicate that the buffer is free to receive more data, the host sends the second 512 byte data block to the memory card. *Id.* at 5:17-18, Fig. 2. After receiving the second and final data block, the memory card again asserts a busy signal on Data0. *Id.* at 5:17-19, Fig. 2. This time, the busy signal indicates "programming busy/ready." *Id.* at 5:20, Fig. 2.

VI. SUMMARY OF THE PROSECUTION HISTORY

The application leading to the '469 Patent was filed on October 14, 2005, and claimed priority to U.S. Provisional Patent Application No. 60/629,098, filed on November 17, 2004. Ex. 1001. The claims as initially drafted recited driving information from a first unit to a second unit over a signal line, driving the signal line to cause a change of state, and interpreting the change of state to have a meaning. Ex. 1002 at 15-21. Other claims were similar, but specifically required an n- block data transfer where, for the first n-1 data blocks, a status signal generated by the second unit was a buffer busy/ready status signal and where a

status signal generated after the nth data block was a programming busy/ready status signal.[3] *Id.*

The Examiner initially rejected the claims in light of MultiMediaCard System Specification Version 3.1 ("MMC 3.1"), an earlier and similar version of MMC 3.31. Ex. 1002 at 210. Important to this Petition, the Examiner noted that MMC 3.1 discloses a multiple block write command that sets the number of blocks to be transferred beforehand. Ex. 1002 at 211-212. Execution of this command is accompanied by busy signaling. *Id.* at 212. MMC 3.1 states: "If all write buffers are full, and as long as the card is in *Programming State*, the DAT line will be kept low (busy)." *Id*. According to the Examiner, this means that "the busy signal of the MMC Spec performs the function of notifying the host whether or not the card is busy, from both the buffers being full and the card being in a programming state." *Id*.

In response to this rejection, the applicant emphasized that, unlike MMC 3.1, the invention of the '469 Patent requires that the same change of state have two

[3] Additional claims also recited an n-block data transfer and similar buffer busy/ready and programming busy/ready status signaling, with the additional requirement that, after the nth data block, a stop transmission command is sent to the second unit. Ex. 1002 at 15-21. These claims were cancelled after the first office action. *Id.* at 228.

different meanings. *Id.* at 235-237. The applicant argued that the busy signal in MMC 3.1 is asserted only when "all write buffers are full, and as long as the card is in the Programming State."[4] Ex. 1002 at 237-238. Thus, according to the applicant, the busy signal of MMC 3.1 only notifies about buffer status, not about whether the card is programming, and therefore the busy signal of MMC 3.1 does not have two different meanings. *Id.* The applicant did not specifically respond to the Examiner's recognition that MMC 3.1 discloses a multiple block write command in which the number of blocks to be transferred is set before execution of the multiple block write command.

In response to the applicant's arguments and amendments, the Examiner again rejected the claims, this time as obvious over MMC 3.1 in view of U.S. Patent No. 6,977,656 ("Lee"). Ex. 1002 at 248. According to the Examiner, Lee discloses a ready signal that indicates data is ready/valid during a transfer and indicates that the memory system is ready to accept a new access during idle times. *Id*. at 249. According to the Examiner, it would have been obvious to incorporate this multi-purpose ready signal as taught by Lee into MMC 3.1 so that the memory

[4] While some of the claims as initially drafted did not explicitly require that the first and second changes of state have different meanings, the applicant amended those claims to require that the first change of state and second change of state have different meanings. Ex. 1002 at 223-225.

can inform the host when it is ready for the next block of data or a new data transfer. *Id.* at 249-251.

In response, the applicant amended the claims and argued that the claims were amended to more clearly recite that "the transition in meaning of the busy signal occurs during an information transmission operation wherein the meaning of the busy signal changes from a first meaning to a second meaning." *Id.* at 264-272, 276, 277. The applicant thus argued that MMC 3.1 in view of Lee does not render the claims obvious because the change in meaning in Lee does not occur during an information transmission operation, but between two distinct operating modes: during a transfer and during idle cycles. Ex. 1002 at 277.

The Examiner yet again rejected the claims. *Id.* at 285-297. The Examiner disagreed with the applicant's argument regarding Lee, stating that "Lee discloses a ready/busy signal that may take on multiple meanings" and maintained that the claims are obvious over MMC 3.1 and Lee. *Id.* at 287-289.

Ultimately, after an unsuccessful pre-appeal request, the Examiner agreed to allow the claims upon Applicant's amendment of all independent claims to recite that busy signaling occurs "during the command execution," and specifically require that the command that initiates an n-block data transfer be a multiple-block transfer command. *Id.* at 308, 311-316, 322-325. As amended, the claims of the

'469 Patent require that the change of state and busy signaling occur during the execution of the same command or within the execution of the same command. *Id.* at 317-319.

VII. CLAIM CONSTRUCTION

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 '469 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 '469 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 and 2.

Under the *Phillips* standard and for clarity, Petitioner provides the following specific construction.[5]

A. "within [a/the] command execution"

Claim 19 recites "within [a/the] command execution." Petitioner proposes construing this term as "while performing in accordance with [a/the] same command," consistent with the prosecution history of the '469 Patent. Ex. 1015 at ¶109. Specifically, the applicant amended Claim 19 to recite "within [a/the] command execution" in a response of April 21, 2009, and argued this to mean within the "same command." Ex. 1002 at 317-319. Further, in the related matter

[5] 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 '469 Patent are invalid under 35 U.S.C. §112 in the co-pending litigation, despite offering explicit and implicit claim constructions herein.

identified above, both Petitioner and Patent Owner agreed that "within [a/the] command execution" in Claim 19 should be construed as "while performing in accordance with [a/the] same command." Ex. 1018 at 3. Accordingly, Petitioner proposes construing "within [a/the] command execution" as "while performing in accordance with [a/the] same command."

VIII. A PERSON OF ORDINARY SKILL IN THE ART

A person of ordinary skill in the art ("POSA") with respect to the technology described in the '469 Patent would be a person with a bachelor's degree in electrical engineering or a closely related field, and two to three years of experience in the field of memory system design. Ex. 1015 at ¶72. However, a higher level of education could make up for less experience, and vice versa. *Id*.

IX. DESCRIPTION OF THE PRIOR ART

A. The MultiMediaCard System Specification, Version 3.31 (Ex. 1003)

MMC 3.31 defines specifications for MultiMediaCards and the communications between MultiMediaCards and their hosts. Ex. 1015 at ¶111. MMC cards are removable electronic cards that can be used as memory devices in portable computers and electronic devices. Ex. 1003 at 11, 15. The MMC specification was well-known to those of skill in the art at the time of the invention of the '469 Patent. Ex. 1015 at ¶112. The MMC specification was used for

commercially available memory cards, with which a POSA would have been familiar. Ex. 1015 at ¶112; Ex. 1007.

In an MMC system, there is typically a host and at least one detachable memory card connected to the host. *See* Ex. 1003 at 11, 15, Fig. 1; Ex. 1015 at ¶113. The card communicates with the host over a bus via three signals: a command line, a data line, and a clock signal. Ex. 1003 at 18. The command line (CMD) "is a bidirectional command channel used for card initialization and data transfer commands." *Id*. "Commands are sent from the MMC bus master to the card and responses from the cards to the host." *Id*. The data line (DAT) "is a bidirectional data channel." *Id*. "Only one card or the host is driving this signal at a time." *Id*. The clock signal is used to synchronize each bit of data transferred on the CMD and DAT signal lines. *Id.*

MMC 3.31 supports block write operations wherein a host will write one or more blocks of data to the card. Ex. 1003 at 38; Ex. 1015 at ¶114. MMC 3.31 discloses two separate multiple block write commands: (1) an "Open-ended Multiple block write," and (2) a "Multiple block write with pre-defined block count." Ex. 1003 at 38. Unlike earlier versions of the MMC specification (prior to MMC 3.1), MMC 3.31 includes a set block count command (CMD23) that allows for a multiple block write command in which the number of blocks to be

transferred is set beforehand. *Id.* at 38, 49. The set block count command "[d]efines the number of blocks which are going to be transferred in the immediately succeeding multiple block read or write command." *Id.* at 49, Table 9.

Using the multiple block write command (CMD25), hosts initiate the transfer of multiple blocks of data to the MMC card. *Id.* at 21, 63 ("In multiple block write mode, the card expects continuous flow of data blocks following the initial host write command."), Fig. 10, Table 9. The card receives these blocks of data in one or more buffers. *Id.* at 35 ("The card may provide buffering for stream and block write. This means that the next block can be sent to the card while the previous is being programmed.")). The card asserts a busy signal on the DAT0 pin to signal to the host when write buffers are full and when the card is in a programming state. *Id.* at 22, 35.

When an open-ended multiple block write is used, the host issues the stop command (CMD12) to terminate the data transfer. Ex. 1003 at 63 ("The data flow is terminated by a stop transmission command (CMD12).")). The host may issue the stop command while the card is programming a data block or while the card is idle. *Id.* at 64, Figs. 33-34. In either case, the card sets a busy signal after

receiving the stop command. *Id.*; Ex. 1015 at ¶116. Figure 10 from MMC 3.31 illustrates the use of busy signals during multiple block write operations.

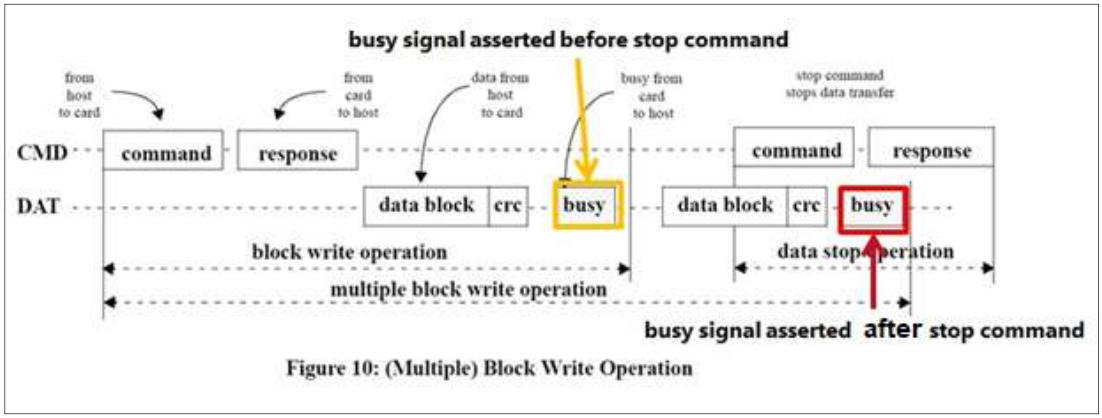


Figure 10: (Multiple) Block Write Operation

As seen in Fig. 10 above, MMC 3.31 discloses that the busy signal asserted ***before*** receipt of the stop command (highlighted in yellow) means that the card's buffers are full. Ex. 1003 at 35. MMC 3.31 also discloses that the busy signal asserted ***after*** receipt of the stop command (highlighted in red) has a different meaning than the previous busy signal. Ex. 1003 at 35 ("As soon as the data transfer is completed, the card will exit the data write state and move either to the *Programming State* (transfer is successful) or *Transfer State* (transfer failed)"); Ex. 1015 at ¶117. When a card executing an open-ended multiple block write command receives a stop command, the card sets a busy signal to indicate that the card is programming, regardless of whether the buffers are full. Ex. 1015 at ¶117. For example, MMC 3.31 includes Figure 34, which the specification describes as showing card behavior when the card receives a stop command while it is idle.

Ex. 1003 at 64. Table 18 explains that the "Z" shown in the timing diagrams depict a "High Impedance State (-> = '1')" in which no data, or command, is being transferred. *Id.* at 59, Table 18; Ex. 1015 at ¶117. This situation is referred to as a "Z-bit." *Id*. at 60; Ex. 1015 at ¶117. The command or data outputs of the card are driven high through external resistors. *Id*. at 60, Fig. 35; Ex. 1015 at ¶117.

MMC 3.31 explains that when a stop command is received in Fig. 34, the card is idle, thereby suggesting that the "Z" depict a state where no data is transferred. Ex. 1003 at 64 ("The following two diagrams describe a scenario of receiving the stop transmission between data blocks. In the first example the card is busy programming the last block while in the second the card is idle."); Ex. 1015 at ¶117. The same symbols (i.e., "Z") are used in Figs. 21-34, confirming that the "Z" means no data is being transferred and the busy signal is not being asserted at that time. *Id.* at 60-64, Figs. 21-34; Ex. 1015 at ¶117.

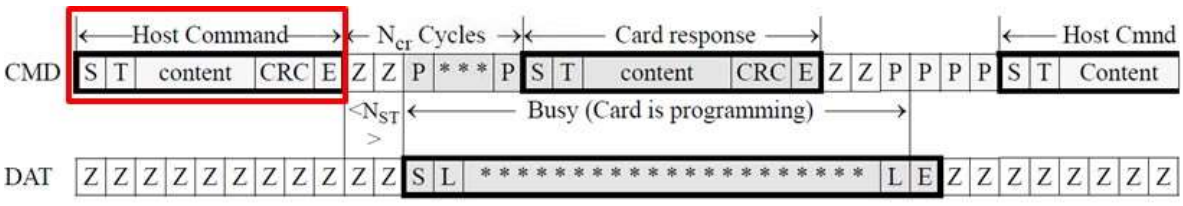


Figure 34: Stop Transmission After Last Data Block. Card Becomes Busy.

Before the stop command (highlighted in red) is received, the "Z" shows that the card is not asserting a busy signal, indicating that the buffers are not full

immediately before the card receives the stop command. Ex. 1015 at ¶118. The card does have unprogrammed data blocks in the buffers and, following the stop command, the figure shows that the card sets a busy signal to indicate that the card is programming. Ex. 1003 at 64, Fig. 34; Ex. 1015 at ¶118.

MMC 3.31 explains that the unprogrammed "blocks are being programmed as soon as the stop transmission command is received and the card activates the busy signal." Ex. 1003 at 64. Thus, unlike the busy signal sent before the stop command is received, this busy signal indicates that the card is programming. Ex. 1015 at ¶117. In other words, the busy signal has a meaning different than the previous buffer busy/ready signals. *Id.* The examiner did not raise this ground of rejection during prosecution of the '469 Patent.

Because the busy signal of MMC 3.31 has a different meaning after a stop command than all previous busy signals, the only potential difference between MMC 3.31 and the '469 Patent is the requirement in the '469 Patent that the signal after the last data block occur within the same command execution as the earlier busier signals. Ex. 1015 at ¶120. Figure 10, highlighted below, demonstrates this difference.

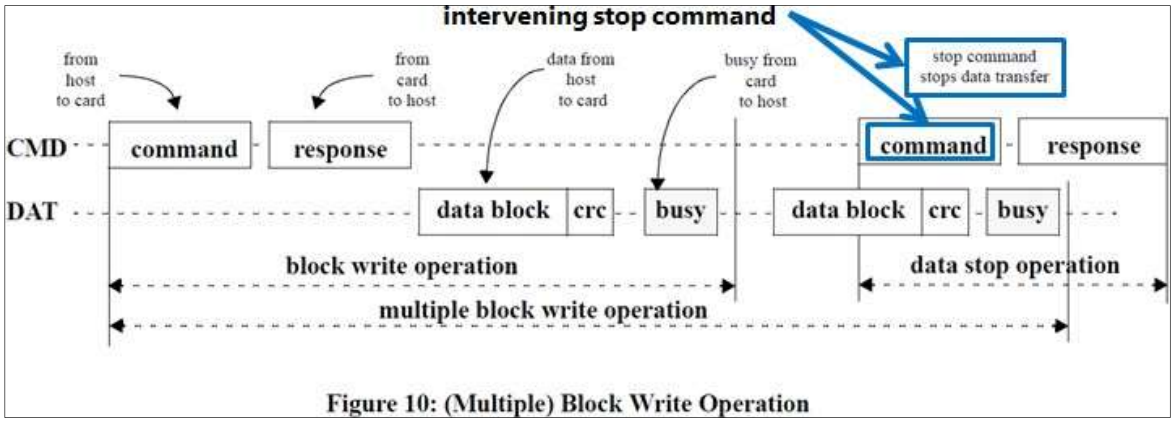


Figure 10: (Multiple) Block Write Operation

The highlighting in blue shows that, when an open-ended multiple block write command is used, there is an intervening stop command before the card provides the busy signal indicating programming. Ex. 1015 at ¶121.

When a pre-defined multiple block write is used, the memory card knows that the data transfer ends when the set number of blocks is received, and without the need for a stop command. Ex. 1003 at 38 ("Multiple block write with pre-defined block count" – "The card will accept the requested number of data blocks, terminate the transaction and return to *transfer* state. Stop command is not required at the end of this type of multiple block write.")). MMC 3.31 discloses that "the host can use either one" of the "[t]wo types of multiple block write transactions" "at any time." *Id*. The stop command may still be used within a pre-defined multiple block write. *Id.* (describing the use of a stop command to abort the write operation either by choice or because of a write error)). However, a card

will respond to a stop command as an illegal command if the command is received after the last block of a pre-defined multiple block write. *Id.*

MMC 3.31 does not include a timing diagram showing that a busy signal is asserted after the card receives the last block of data during a pre-defined multiple block write. However, MMC 3.31 defines the open-ended multiple block write operation and the pre-defined multiple block write in the same section, and does not indicate any difference between the two operations other than the omission of the stop command. Ex. 1003 at 34-35, 38. The state diagram, reproduced below, confirms that both multiple block write operations work the same way. Ex. 1015 at ¶124. The multiple block write command (CMD 25) puts the MMC card into the "Receive- data State (rcv)" (highlighted in blue), during which the card receives blocks of data. The card can leave the receive state in one of two ways, as indicated by the box highlighted in red, "CMD12 [i.e. the stop command] or transfer end."

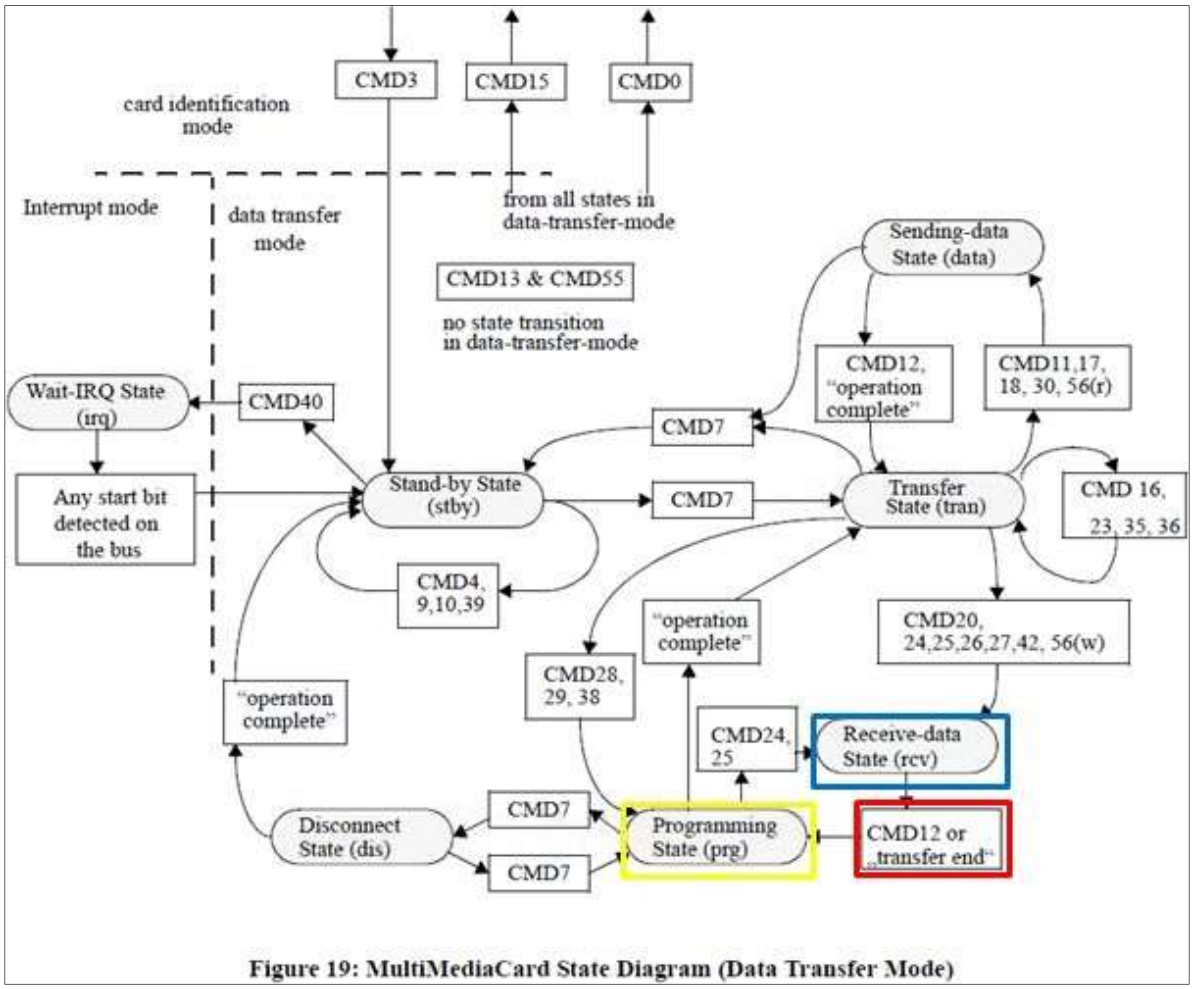


Figure 19: MultiMediaCard State Diagram (Data Transfer Mode)

This transition puts the card into "Programming State (prg)" (highlighted in yellow), where the card asserts the busy signal. Thus, the busy signaling during a multiple block write operation, as described in Fig. 10 and Fig. 34 above, is unchanged whether the stop command is used or omitted. Ex. 1015 at ¶124.

MMC 3.31 appears on the face of the '469 Patent as a reference cited during prosecution. Ex. 1001 (References Cited, Other Publications).

MMC 3.31 bears a copyright date of May 2003 and was publicly available on the MultiMediaCard Association ("MMCA") website, www.mmca.org, at least as of June 3, 2003. Ex. 1003 at 3; Ex. 1005 at 5. As of November 22, 2001, the MMCA website homepage contained a link for host platform developers to buy an MMCA specification. Ex. 1005 at 11. As of June 3, 2003, the linked page stated that MMC 3.31 was available for purchase. Ex. 1005 at 5. This page provided links to a "System Summary," a "Table of Contents of the MMCA System Specification," and a link to place an order for MMC 3.31. *Id*. The linked System Summary document bears a copyright date of March 2003, and was available on the MMCA website as of at least June 29, 2003. Ex. 1005 at 14-51. As of August 18, 2003, the link to place an order redirected you to a form requesting "Customer Information" and "Product Interest" that would be submitted with the order for MMC 3.31. Ex. 1005 at 6.

MMC 3.31 is therefore prior art under § 102(b).

B. CompactFlash (Ex. 1004)

CompactFlash Specification Revision 1.3 (Ex. 1004, "CompactFlash") defines the specifications for CompactFlash Storage Cards and communications between CompactFlash cards and their hosts. Ex. 1004 at 7; Ex. 1015 at ¶129. CompactFlash cards are intended to be high capacity storage cards, and generally

comply with the Personal Computer Memory Card International Association ATA standard ("ATA"), a widely used specification in disk and other storage devices. Ex. 1004 at 7, 8.

The CompactFlash specification was well-known to a POSA at the time of the invention of the '469 Patent. Ex. 1015 at ¶131. Indeed, the CompactFlash specification was used for commercially available memory cards, with which a POSA would have been familiar. Ex. 1009; Ex. 1015 at ¶131.

Similar to the commands disclosed by MMC 3.31, CompactFlash discloses commands for transferring multiple portions, e.g. sectors, of data to a card using a command. Ex. 1015 at ¶131. For example, CompactFlash discloses a Write Sector(s) command that "writes from 1 to 256 sectors" of data. Ex. 1004 at 80.

These sectors are similar to the blocks of MMC 3.31. Ex. 1015 at ¶133. The Write Sector(s) command, like the open-ended multiple block write command of MMC 3.31, signals to the host when the card's buffer is ready to receive additional data by driving a change of state on an interrupt pin. *Id.*; Ex. 1015 at ¶134. After receiving each sector, the card sets a busy bit in the Status Register. Ex. 1004 at 53, 55, 80. When the card buffer is ready to receive the next sector, the busy bit is cleared and the interrupt signal is generated. *Id.* This interrupt is a

change of state that signals to the host that the buffer can receive additional data. *Id*.

Execution of the write sectors command does not require a stop command to stop the data transfer, but writes the number of "sectors as specified in the Sector Count Register." Ex. 1004 at 80. After the final sector is transferred, a busy bit is again set in the Status Register. *Id*. When the command is completed (i.e., the data has been programmed), the busy bit is cleared and the card generates the same interrupt. *Id.*; Ex. 1015 at ¶136. This final interrupt signal, like the interrupts between sectors, is a change of state that communicates status to the host. Ex. 1015 at ¶136. In this case, the meaning of the change of state is that programming is complete. Ex. 1004 at 80; Ex. 1015 at ¶¶136, 137.

CompactFlash Specification Revision 1.3 was publicly available prior to 2000. Ex. 1006 at ¶¶5-6 (Declaration of Michael Asao, consultant for the CompactFlash Association ("CFA")). Any interested member of the public could have visited the CFA website and downloaded a copy of the current standard directly from the website. Ex. 1006 at ¶4.

The Wayback Machine maintained by Internet Archive at archive.org confirms that CompactFlash was available from the CFA website as of at least December 1998. Ex. 1005 at 57, 58. The CFA website homepage included a link

entitled, "CF Spec. Ver. 1.3 Free Download." *Id.* at 57. The linked page was entitled, "Registration for CompactFlash™ Specification Download," and requested registration information "to download the CF Specification." Ex. 1005 at 58. After filling out this registration information and submitting the form, the CompactFlash specification could be downloaded directly from the CFA website. Ex. 1006 at ¶¶4-6.

CompactFlash is therefore prior art under § 102(b).

X. GROUND 1: CLAIMS 19 and 20 ARE OBVIOUS IN VIEW OF MMC 3.31.

As shown below, MMC 3.31 renders Claims 19 and 20 of the '469 Patent obvious under 35 U.S.C. § 103. *See also* Ex. 1015 at ¶¶138-175.

A. Claim 19

1. Claim 19 [preamble]

To the extent that the preamble is limiting, MMC 3.31 discloses "[a] memory device." MMC 3.31 discloses a bus that connects a host and a memory card. Ex. 1003 at Fig. 1 (as shown below); Ex. 1015 at ¶138.

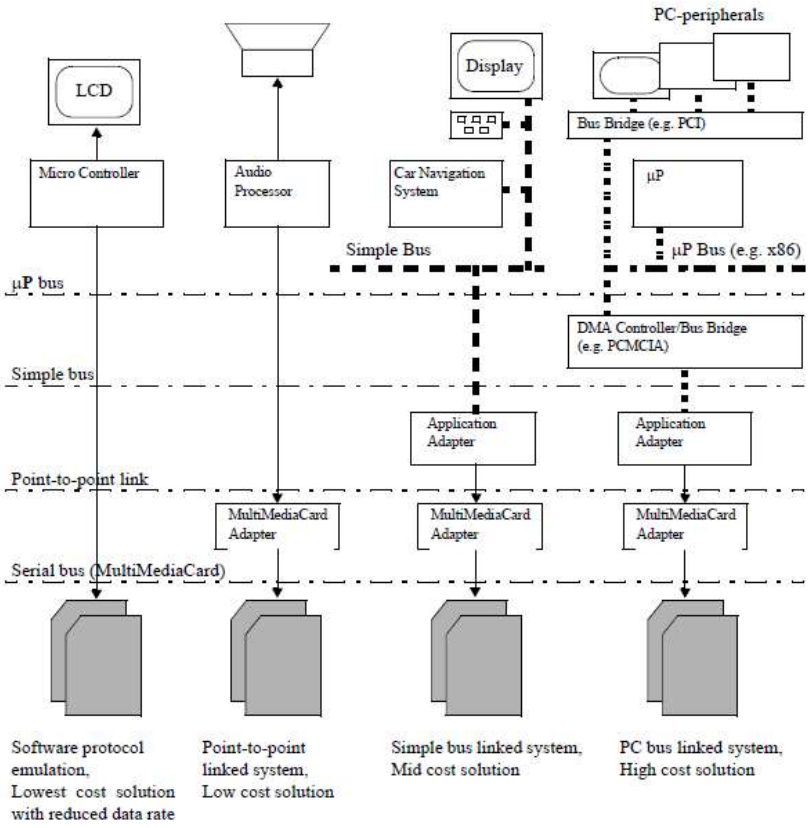


Figure 1: Topology Of MultiMediaCard Systems

Further, as shown below (Fig. 4), MMC 3.31 discloses the specific configuration of a memory card (MMC card). Ex. 1003 at 19, Fig. 4.

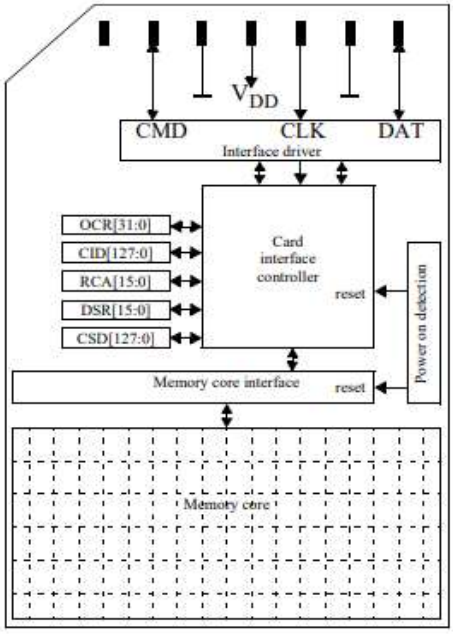


Figure 4: MultiMediaCard Architecture

Therefore, MMC 3.31 discloses a memory device. Ex. 1015 at ¶139.

2. Claim 19[a]

MMC 3.31 discloses "a bus interface configured to be coupled to a host through a bus having a data signal line." As shown in Figure 1 above, MMC 3.31 discloses a host, such as a PC peripheral, connected to a memory card through a bus. Ex. 1003 at Fig. 1. Further, as shown in Figure 4 above, MMC 3.31 discloses an MMC card including a bus interface. Ex. 1003 at 19, Fig. 4; Ex. 1015 at ¶140. The bus interface of the MMC card includes power terminals, a command terminal, a data terminal, and a clock terminal. Ex. 1003 at 18, at 19, Fig. 4; Ex. 1015 at ¶140. The card communicates with the host over a bus via three signals: a command line, a data line, and a clock signal. Ex. 1003 at 18. The command line

(CMD) "is a bidirectional command channel used for card initialization and data transfer commands." *Id.* "Commands are sent from the MMC bus master to the card and responses from the cards to the host." *Id.* The data line (DAT) "is a bidirectional data channel." *Id.* "Only one card or the host is driving this signal at a time." *Id.* The clock signal is used to synchronize each bit of data transferred on the CMD and DAT signal lines. *Id.*; Ex. 1015 at ¶140.

Therefore, MMC 3.31 discloses and renders obvious claim element 19[a]. Ex. 1015 at ¶141.

3. Claim 19[b]

Claim 19[b] recites "the bus interface further comprising a driver at said memory device coupled to said data signal line and a receiver at said memory device coupled to the data signal line, said receiver being operable to receive information comprising a first information portion and a second information portion from the host over the data signal line within a command execution." As described below, the MMC 3.31 discloses and renders obvious this limitation. Ex. 1015 at ¶¶142-146.

MMC 3.31 discloses that the "CMD signal has two operation modes: open-drain for initialization mode and push-pull for fast command transfer" and that the "DAT signal operates in push-pull mode." Ex. 1003 at 18. A POSA would understand that open-drain and push-pull refer to types of drivers. Ex. 1015 at ¶143. In addition, Figure 39 (as shown below) of MMC 3.31 illustrates the driver and receiver located in each of the memory cards. Ex. 1003 at Fig. 39 (annotated below).

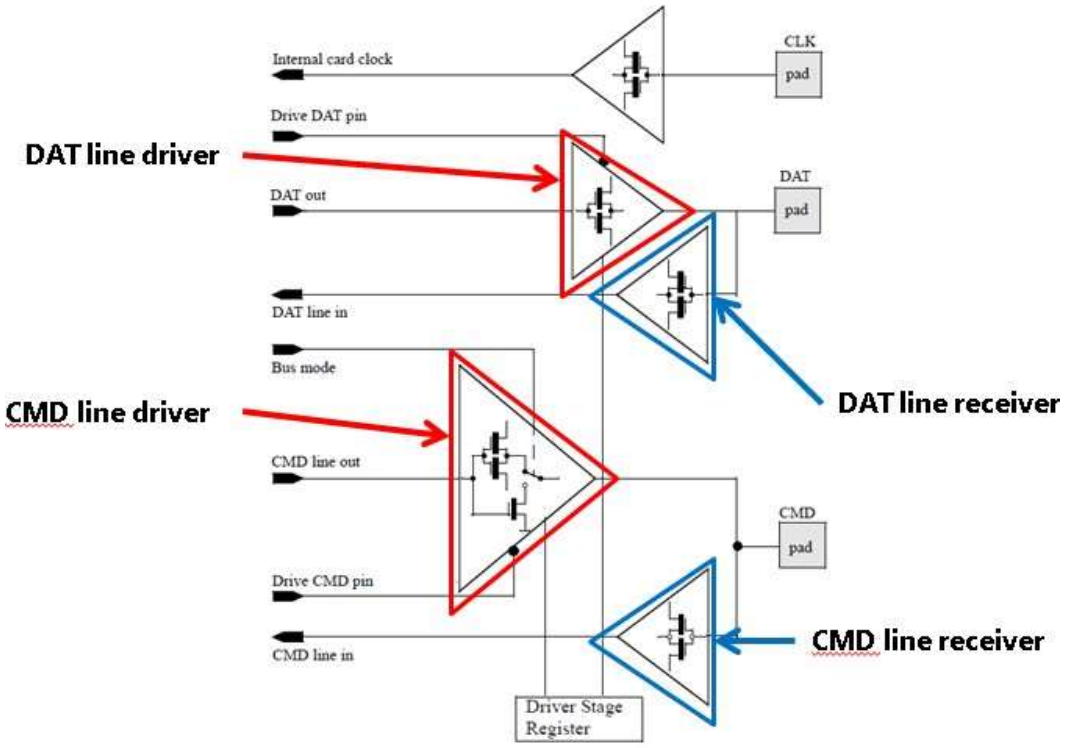


Figure 39: MultiMediaCard Bus Driver

As illustrated above, the data line (i.e., DAT out) is connected to a driver and the data line (i.e., DAT line in) is connected to a receiver. Further, MMC 3.31 describes that the bus interface of the MMC card includes the above-noted driver and receiver in Figure 39. Ex. 1003 at 82 ("The CMD and DAT bus drivers consist of a predriver stage and a complementary driver transistor"). At least based on the description that the "CMD signal has two operation modes: open- drain for initialization mode and push-pull for fast command transfer" and that the "DAT signal operates in push-pull mode," illustration in Figure 39 of the DAT line driver and the DAT line receiver, and description on page 82 of the MMC bus drivers, a POSA would understand that MMC 3.31 discloses that "the bus interface further comprising a driver at said memory device coupled to said data signal line and a receiver at said memory device coupled to the data signal line." Ex. 1015 at ¶144.

Further, MMC 3.31 discloses that the CMD and DAT lines are bidirectional data channels. Ex. 1003 at 18 ("CMD: is a bidirectional command channel used for card initialization and data transfer commands."); *id.* ("DAT: is a bidirectional data channel."). Based on this description, a POSA would understand that the memory card (for example, DAT line receiver and CMD line receiver noted above in Figure 39) in MMC 3.31 can receive information over the CMD and DAT lines from the host. Ex. 1015 at ¶145.

Additionally, MMC 3.31 describes that the received information can comprise a first information portion and a second information portion, such as when executing a single multiple block write command, in which case the multiple data blocks are multiple "information portions." Ex. 1003 at 38, Fig. 10. Specifically, MMC 3.31 provides a set block count command that allows for a multiple block write command to terminate after a pre-defined number of blocks are transferred. *Id.* at 38. MMC 3.31's pre-defined number of data blocks that are transferred from a host to a card (while performing in accordance with the same multiple block write command CMD25) are the claimed "first information portion" and "second information portion" received "within a command execution." *Id.* at 38; Ex. 1015 at ¶146. *See supra*, VII.A.[6] Accordingly, based on the description of multiple-write block with pre-defined block count in MMC 3.31, a POSA would understand that MMC 3.31 discloses "said receiver being operable to receive information comprising a first information portion and a second information portion from the host over the data signal line within a command execution." Ex. 1015 at ¶146.

[6] For the purposes of this proceeding, Petitioner is applying Patent Owner's construction from the related litigation matter of "[first/second] information portion" to mean "a [first/second, or another], portion of the information." Ex. 1018 at 17. Petitioner reserves the right to argue for a different construction in the related litigation matter.

Therefore, MMC 3.31 discloses and renders obvious claim element 19[b]. Ex. 1015 at ¶146.

4. Claim 19[c]

Claim 19[c] recites "said driver being operable to drive a change of state of the data signal line to the host within the command execution." As described below, MMC 3.31 discloses and renders obvious claim element 19[c]. Ex. 1015 at ¶¶147-153.

After receiving one or more information portions (i.e., one or more data blocks) on the DAT line in response to a write multiple command (for example, CMD25), the memory card asserts a busy signal (from card to host) on the DAT line by pulling the signal low.

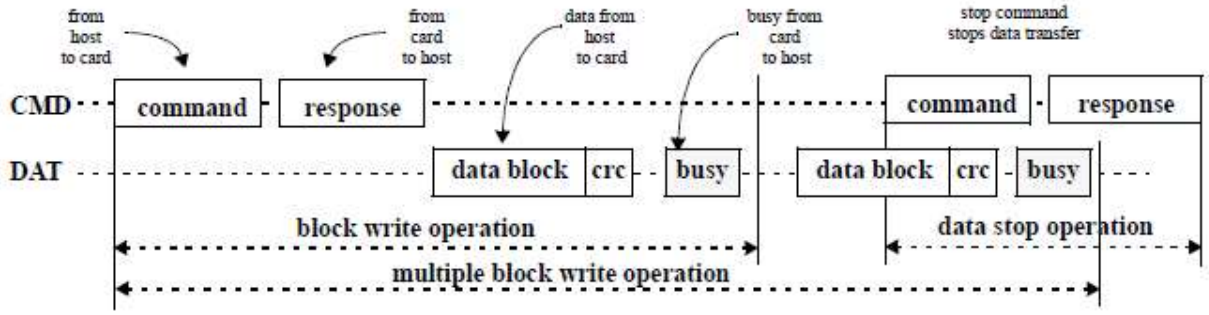


Figure 10: (Multiple) Block Write Operation

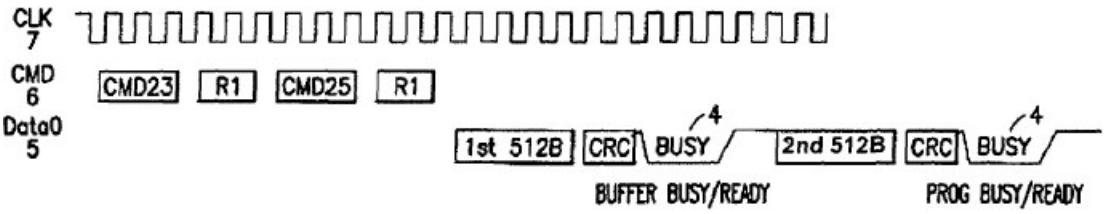
Ex. 1003 at 22, 35, 38, Fig. 10 (shown above). Once the card has a buffer that is not full, the host de-asserts the busy signal on the DAT line. *See id.*

MMC 3.31 discloses this scenario as one of two instances in which the busy signal is asserted by the memory card. Ex. 1003 at 35 ("If all write buffers are full . . . the DAT line will be kept low."). This change of state of the busy signal occurs during execution of the write multiple block command (CMD25). *See* Ex. 1003 at Figs. 10. Since MMC 3.31 discloses transferring a plurality of data blocks after a write multiple block command (CMD25) during a multiple block write with pre-defined block transaction, a POSA would understand that the memory card asserts a busy signal on the DAT line after a receiving each of the plurality of data block. Ex. 1003 at 38; Ex. 1015 at ¶149. This busy signal is the "change of state of the data signal line." Ex. 1015 at ¶149.

In the context of the '469 Patent, "within the command execution" refers to the time between initiation of the command and initiation of the next command (i.e., while performing in accordance with the same command). The specification of the '469 Patent explains that "in accordance with aspects of this invention the meaning of the busy signal 4 is changed within the same command (e.g., the multiple block write) and between transferred data blocks."

Ex. 1001 at 5:26-29. Figure 2 of the '469 Patent is illustrative, and shows the busy signals being asserted after initiation of the multiple block write command (CMD25), but before any other command. Ex. 1001 at Fig. 2.

FIG.2



Similarly, in response to the last office action before the Examiner allowed the application, the applicant explained that the busy signals within the scope of the '469 Patent occur between blocks of a multi-block data transfer: "In exemplary embodiments detailed in the specification, for all but the last block of a multi-block data transfer the ready signal is interpreted as *buffer busy/ready*, whereas for the last data block in that same multi-block data transfer the ready signal is interpreted as *programming busy/ready*." Ex. 1002 at 319 (emphasis added).

Thus, because the busy signal of MMC 3.31 occurs after a data block is received while performing in accordance with the same write multiple block command (i.e., CMD25) and before any other command, the change of state of the data signal line to the host is "within the command execution" (i.e., while

performing in accordance with the same multiple block write command CMD25). Ex. 1015 at ¶151.

Further, as discussed above, MMC 3.31 discloses that the memory card contains a driver which is connected to DAT out terminal. *See supra* X.A.3 (Claim 19[b]). A POSA would understand that MMC 3.31's driver (which is connected to DAT out) is operable to drive a change of state of the busy signal on the data signal line to the host. Ex. 1003 at 82 ("The CMD and DAT bus drivers consist of a predriver stage and a complementary driver transistor (Figure 39). The predriver stage output rise and fall time is set with the DSR1 register and determines the speed of the driver stage. The complementary driver transistor size is set with the DSR2 register and determines the current driving capabilities of the driver stage and also influences the peak current consumption of the bus driver. The proper combination of both allows the optimum bus performance."); Ex. 1015 at ¶152.

Therefore, MMC 3.31 discloses and renders obvious claim element 19[c]. Ex. 1015 at ¶153.

5. Claim 19[d]

Claim 19[d] recites that "said bus interface further comprising a controller coupled to said driver and to said receiver and operable to cause the change of state of the data signal line to have a first meaning after receiving the first information

portion within the command execution and to have a second meaning different from the first meaning after receiving the second information portion within the command execution from the host over the data signal line."[7] As described below, MMC 3.31 discloses and renders obvious claim element 19[d]. Ex. 1015 at ¶¶154-173.

MMC 3.31 discloses that the memory card contains a controller coupled to an interface driver. Ex. 1003 at 19, Fig. 4 (shown below). A POSA would understand that this controller is part of the bus interface and implements the communication protocol disclosed by MMC 3.31, including how the host should interpret signals from the memory card. Ex. 1015 at ¶155.

[7] As described in the '469 Patent, the controller of the memory card causes the same change of state of the signal line after receiving the first and second information portions. Ex. 1001 at 1:29-31, 5:14-20, Fig. 2. Thus, because the same signal is sent, there is no disclosure in the '469 Patent that the memory card controller causes the change of state to have different meaning. Instead, the host device receives an identical change of state from the card, and the host, not the card, interprets the change of state to have a different meaning. Ex. 1001 at 4:4-10, 4:56-58. Petitioner addresses this claim element consistent with the disclosure of the '469 Patent, and reserves its right to challenge the validity of this claim under Section 112 in other fora.

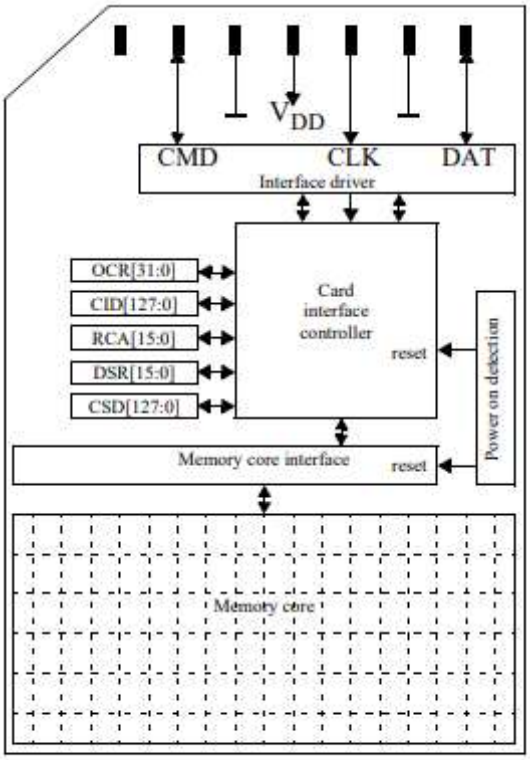


Figure 4: MultiMediaCard Architecture

Further, MMC 3.31 supports block write operations, wherein a host will write one or more blocks of data to the card. Ex. 1003 at 38; Ex. 1015 at ¶156. MMC 3.31 discloses two separate multiple block write commands: (1) an "Open-ended Multiple block write," and (2) a "Multiple block write with pre-defined block count." Ex. 1003 at 38. Unlike earlier versions of the MMC specification (prior to MMC 3.1), MMC 3.31 includes a set block count command (CMD23) that allows for a multiple block write command in which the number of blocks to be transferred is set beforehand. *Id.* at 38, 49. The set block count command "[d]efines the number of blocks which are going to be transferred in the

immediately succeeding multiple block read or write command." *Id.* at 49, Table 9.

Using the multiple block write command (CMD25), hosts initiate the transfer of multiple blocks of data to the MMC card. *Id.* at 21, 63 ("In multiple block write mode, the card expects continuous flow of data blocks following the initial host write command."), *Id.* at 22, Fig. 10; *Id.* at 49, Table 9. The card receives these blocks of data in one or more buffers. *Id.* at 35 ("The card may provide buffering for stream and block write. This means that the next block can be sent to the card while the previous is being programmed."). The card asserts a busy signal on the DAT0 pin to signal to the host when write buffers are full and when the card is in a programming state. *Id.* at 22, 35.

When an open-ended multiple block write is used, the host issues the stop command (CMD12) to terminate the data transfer. Ex. 1003 at 63 ("The data flow is terminated by a stop transmission command (CMD12)."). The host may issue the stop command while the card is programming a data block or while the card is idle. *Id.* at 64, Figs. 33-34. In either case, the card sets a busy signal after receiving the stop command. *Id.*; Ex. 1015 at ¶158. Figure 10 from MMC 3.31 illustrates the use of busy signals during multiple block write operations.

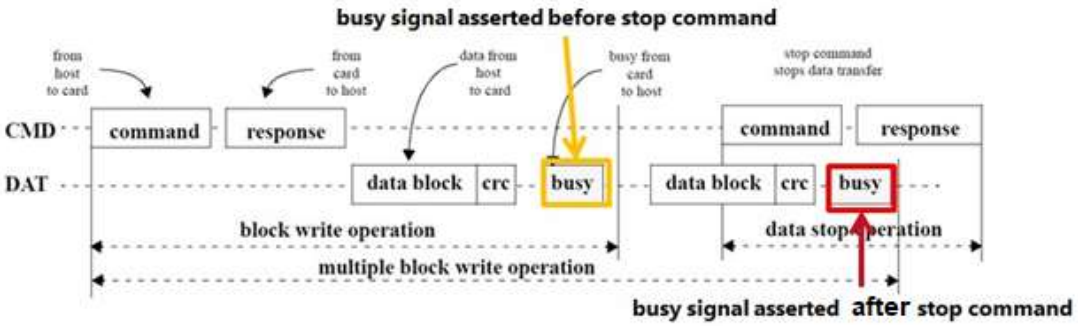


Figure 10: (Multiple) Block Write Operation

As seen in Fig. 10 above, MMC 3.31 discloses that the busy signal asserted ***before*** receipt of the stop command (highlighted in yellow) means that the card's buffers are full. Ex. 1003 at 35. MMC 3.31 also discloses that the busy signal asserted ***after*** receipt of the stop command (highlighted in red) has a different meaning than the previous busy signal. Ex. 1003 at 35 ("As soon as the data transfer is completed, the card will exit the data write state and move either to the *Programming State* (transfer is successful) or *Transfer State* (transfer failed)"; Ex. 1015 at ¶159.

When a card executing an open-ended multiple block write command receives a stop command, the card sets a busy signal to indicate that the card is programming, regardless of whether the buffers are full. Ex. 1015 at ¶160. For example, MMC 3.31 includes Figure 34, which the specification describes as showing card behavior when the card receives a stop command while it is idle. Ex. 1003 at 64. Table 18 explains that the "Z" shown in the timing diagrams

depict a "High Impedance State" in which no data is being transferred. *Id.* at 59, Table 18; Ex. 1015 at ¶159. MMC 3.31 explains that when a stop command is received in Fig. 34, the card is idle, thereby suggesting that the "Z" depicts a state where no data is transferred. Ex. 1003 at 64 ("The following two diagrams describe a scenario of receiving the stop transmission between data blocks. In the first example the card is busy programming the last block while in the second the card is idle."); Ex. 1015 at ¶159. The same symbols (i.e., "Z") are used in Figs. 21-34, confirming that the "Z" means no data is being transferred and the busy signal is not being asserted at that time. *Id.* at 60-64, Fig. 21-34.

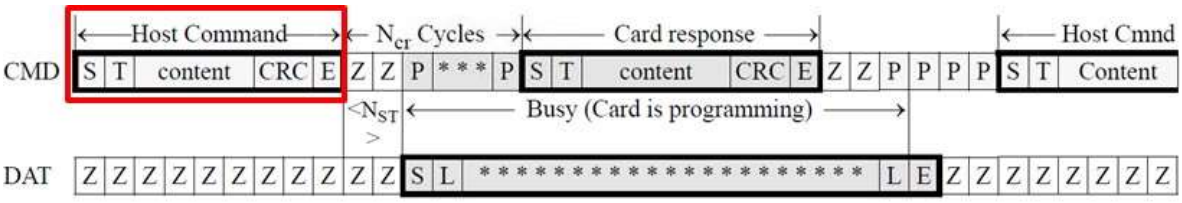


Figure 34: Stop Transmission After Last Data Block. Card Becomes Busy.

Before the stop command (highlighted in red) is received, the "Z" shows that the card is not asserting a busy signal, indicating that the buffers are not full immediately before the card receives the stop command. Ex. 1015 at ¶160. The card does have unprogrammed data blocks in the buffers and, following the stop command, the figure shows that the card sets a busy signal to indicate that the card

is programming. Ex. 1003 at 64, Fig. 34 ("Busy (Card is Programming)"; Ex. 1015 at ¶160.

MMC 3.31 explains that the unprogrammed "blocks are being programmed as soon as the stop transmission command is received and the card activates the busy signal." Ex. 1003 at 64. Thus, unlike the busy signal sent before the stop command is received, *this* busy signal at a different time indicates that the card is programming. Ex. 1015 at ¶161. In other words, the busy signal has a meaning different than the previous buffer busy/ready signals. *Id.* Based on the above-noted description in MMC 3.31, a POSA would have understood that the card uses the busy signal to have two distinct meanings at different times. Ex. 1015 at ¶162.

Because the busy signal of MMC 3.31 has a different meaning after a stop command than all previous busy signals, the only potential difference between MMC 3.31 and the '469 Patent is the requirement in the '469 Patent that the busy signal after the second information portion occur within the same command execution as the earlier busier signals. Ex. 1015 at ¶163. Figure 10, highlighted below, demonstrates this difference.

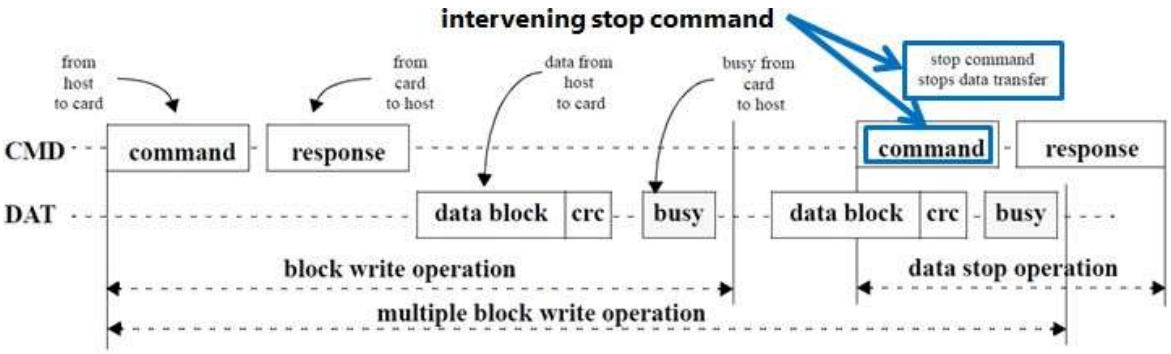


Figure 10: (Multiple) Block Write Operation

The highlighting in blue shows that, when an open-ended multiple block write command is used, there is an intervening stop command before the card provides the busy signal indicating programming. ***However, a pre-defined multiple block write operation in MMC 3.31 does not require an intervening stop command***.

In particular, when a pre-defined multiple block write is used, the memory card knows that the data transfer ends when the set number of blocks is received, and without the need for a stop command. Ex. 1003 at 38 ("Multiple block write with pre- defined block count" – "The card will accept the requested number of data blocks, terminate the transaction and return to *transfer* state. Stop command is not required at the end of this type of multiple block write."). MMC 3.31 discloses that "the host can use either one" of the "[t]wo types of multiple block write transactions" "at any time." *Id*. The stop command may still be used within a pre-defined multiple block write. *Id.* (describing the use of a stop command to

abort the write operation either by choice or because of a write error). However, a card will respond to a stop command as an illegal command if the command is received after the last block of a pre-defined multiple block write. *Id.*

MMC 3.31 does not include a timing diagram showing that a busy signal is asserted after the card receives the last block of data during a pre-defined multiple block write. However, MMC 3.31 defines the open-ended multiple block write operation and the pre-defined multiple block write in the same section, and does not indicate any difference between the two operations other than the omission of the stop command. Ex. 1003 at 34-35, 38. The state diagram, reproduced below, confirms that both multiple block write operations work the same way. The multiple block write command (CMD 25) puts the MMC card into the "Receive-data State (rcv)" (highlighted in blue), during which the card receives blocks of data. The card can leave the receive state in one of two ways, as indicated by the box highlighted in red, "CMD12 [i.e. the stop command] or "transfer end.""

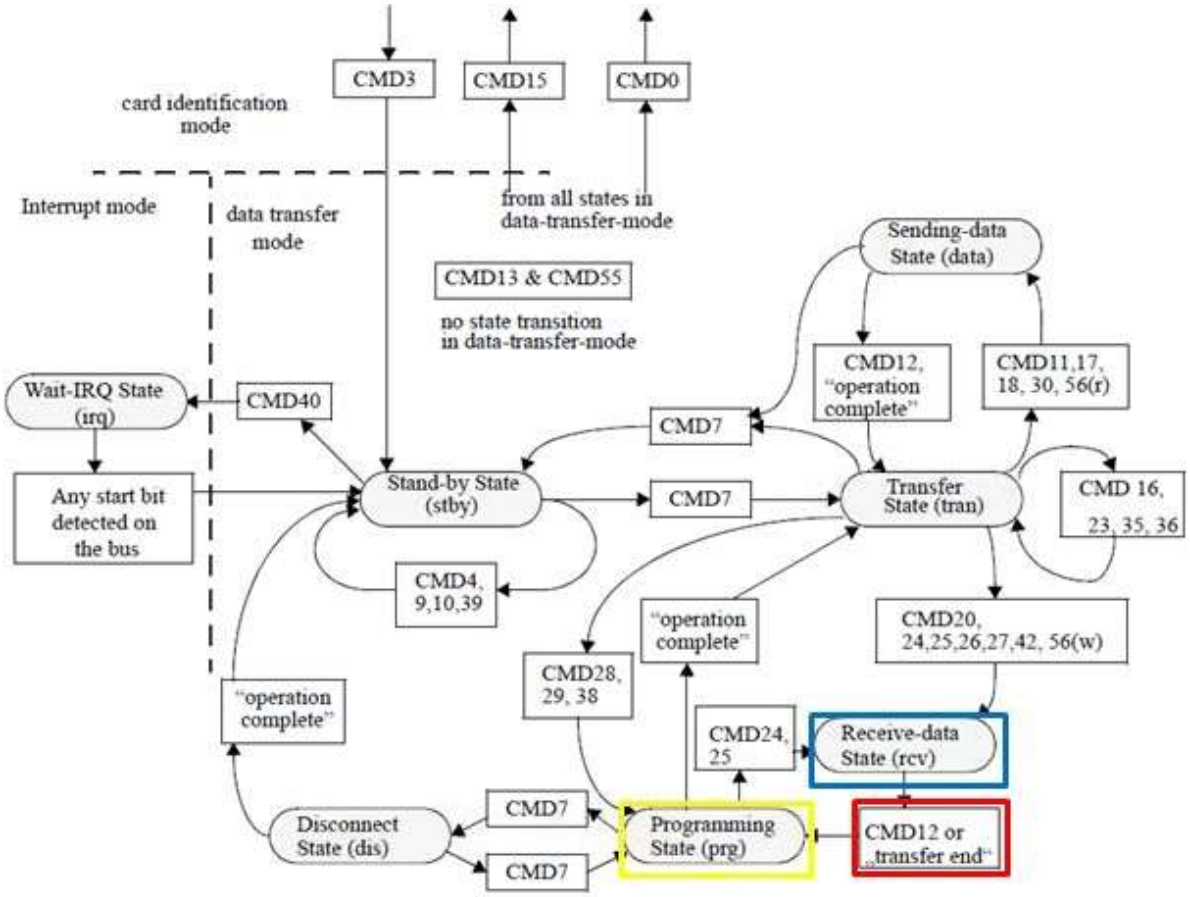


Figure 19: MultiMediaCard State Diagram (Data Transfer Mode)

This transition puts the card into "Programming State (prg)" (highlighted in yellow), where the card asserts the busy signal. Thus, the busy signaling during a multiple block write operation, as described in Fig. 10 and Fig. 34 above, is unchanged whether the stop command is used or omitted. Ex. 1015 at ¶167.

In view of the above state diagram, a POSA would have found it obvious that the card could assert a busy signal following the last block of a pre-defined multiple block write operation to indicate to the host that the card is programming.

Ex. 1015 at ¶168. In this instance, the card is aware that the host is finished transferring blocks because the SET_BLOCK_COUNT command (CMD23) provided this information before the multiple block write operation began. Ex. 1003 at 38. It would be logical to use the same busy signaling implemented within an open-ended multiple block write operation, in which the busy signal after the last block is received indicates programming, because the same state machine is disclosed for both the open-ended and pre-defined multiple block writes. Ex. 1003 at 34-35, Fig. 19; Ex. 1015 at ¶168.

A POSA would have realized that the busy signaling technique that was implemented with respect to an open-ended multiple block write operation would achieve the same benefits if used with the pre-defined multiple block write. Ex. 1015 at ¶169. Using the busy signal would allow the host to know the programming status of the card in both types of multiple block writes. Ex. 1015 at ¶169. In other words, interpreting the busy signal after the transfer ended to mean programming would improve the system in the same way. Ex. 1015 at ¶169. The card could assert the busy signal following the final block of the pre-defined multiple block write when the card buffer is not full, such as in the situation illustrated in Figure 34. Ex. 1003 at 64, Fig. 34; Ex. 1015 at ¶169. In this

situation, the busy signal would have a different meaning than all previous busy signals. Ex. 1015 at ¶169.

A POSA also would have found it obvious that the busy signal could be used to indicate that the memory card is programming because the physical components of the card were specified in MMC 3.31 to provide a programming busy signal when being used in the Serial Peripheral Interface ("SPI") mode. Ex. 1003 at 94; Ex. 1010 at 11:14-19; Ex. 1015 at ¶170. When the card is operating in the SPI mode, after each block of a multiple block transfer is received by the card, the card will send "a continuous stream of busy tokens" to the host "[a]s long as the card is busy programming." Ex. 1003 at 94. Thus, a POSA would have understood that it would not have been necessary to modify the memory card of MMC 3.31 physically to allow for the busy signal to be used to indicate that the card was busy programming. Ex. 1015 at ¶170. A POSA would have realized that the components of the memory card specified in MMC 3.31 could be used to send a busy signal that indicated that the card was programming data to the host. Ex. 1015 at ¶170.

Accordingly, as explained above, the express disclosure of a busy signal with two different meanings in the open-ended multiple block write operation and the description of the state diagram for both multiple block write operations would

have provided an express motivation to a POSA to apply the signaling scheme of the open-ended multiple block write operation to the pre-defined multiple block write operation. Ex. 1015 at ¶171. In other words, a POSA would have understood that the two busy signals (with different meanings) of MMC 3.31 could be implemented in either the open-ended or the pre-defined multiple block write operation. Ex. 1015 at ¶171. A POSA would have been motivated to apply the same busy signaling technique used in the open-ended multiple block write for the pre-defined multiple block write because a POSA would realize that the busy signaling technique that was implemented with respect to an open-ended multiple block write operation would improve the pre-defined multiple block write in the same way. Ex. 1015 at ¶172; *KSR Intern. Co. v. Teleflex Inc.*, 550 U.S. 398, 417 (2007). A POSA would have recognized that the same gain in efficiency could be realized using the same busy signaling with the pre-defined multiple block write operation. Ex. 1015 at ¶172.

Accordingly, since a POSA would be motivated to apply the same busy signaling scheme (i.e., different meanings of the busy signals in the open-ended multiple block write operation) to MMC 3.31's pre-defined multiple block write operation, and since the pre-defined multiple block write operation does not require an intervening stop command (because the memory card knows that the

data transfer ends when the set number of blocks is received), MMC 3.31 discloses causing "the change of state of the data signal line to have a first meaning after receiving the first information portion within the command execution and to have a second meaning different from the first meaning after receiving the second information portion within the command execution from the host over the data signal line." As such, MMC 3.31 discloses or renders obvious claim element 19[d]. Ex. 1015 at ¶173.

B. Claim 20

Claim 20 depends from Claim 19, and requires that the bus is comprised of a command signal line and that the controller is responsive to at least one command for a multi-block transfer that initiates the command execution received through the command signal line from the host. MMC 3.31 discloses that all data transfer commands are sent over the CMD line, which is separate from the DAT line. Ex. 1003 at 18, 21-22. The multiple block write command is sent over the CMD line and causes multiple blocks to be written to the memory card. Ex. 1003 at Fig. 10.

As noted above, MMC 3.31 discloses that the memory card contains a controller coupled to an interface driver. Ex. 1003 at 20, Fig. 4. A POSA would understand that this controller is part of the bus interface and implements the communication protocol disclosed by MMC 3.31, including how the host should

interpret command signals from the memory card. Ex. 1003 at 14; Ex. 1015 at ¶174.

Further, as discussed with respect to Claim 19, MMC 3.31 discloses that when executing the multiple block write command, the change of state of the data signal line is caused to have a first meaning after receiving the first data block, and to have a second meaning after receiving the second data block. *See supra* Section X.A.5, Claim 19, e.g., Claim 19[d]; *see also* Ex. 1015 at ¶175.

Accordingly, MMC 3.31 discloses and renders obvious Claim 20. Ex. 1015 at ¶175.

XI. GROUND 2: CLAIMS 19 AND 20 ARE OBVIOUS IN VIEW OF MMC 3.31 AND COMPACTFLASH.

As shown below, MMC 3.31 and CompactFlash render Claims 19 and 20 of the '469 Patent obvious under 35 U.S.C. § 103. *See also* Ex. 1015 at ¶¶ 176-203.

A POSA would have been motivated to combine MMC 3.31 and CompactFlash. A POSA looking to improve on MMC 3.31 would have been motivated to combine MMC 3.31 and CompactFlash. Ex. 1015 at ¶176. As discussed above, MMC 3.31 discloses a removable memory card that can be used with computer devices. *Supra*, IX.A. Stated goals of the MMC standard were high mobility, high performance, low price, low power consumption, and high data throughput at the memory card interface. Ex. 1003 at 11.

MMC 3.31 provides commands for a host to write blocks of data to the card, including a multiple block write command. *Id.* at 38, Fig. 10. MMC 3.31 provides a set block count command that allows for a multiple block write command to terminate after a pre-defined number of blocks are transferred. *Id.* at 38. The multiple block write commands further the aims of high throughput, while reducing the overhead that would be associated with the sequential use of multiple, single write block commands. Ex. 1015 at ¶176. MMC 3.31 discloses using busy signals on the DAT line with two different meanings: one to indicate buffer busy/ready after each block of data is received, and a second to indicate programming after the stop command is received. *Supra*, X.A.5 (Claim 19[d]); Ex. 1003 at 34-35, 38; Ex. 1015 at ¶176.

To the extent the host might not interpret the second signal to indicate programming, a POSA would have been motivated to apply the busy signaling scheme of CompactFlash, which expressly signals buffer ready and programming complete so that the host would know the status of the card during multiple sector write operations. Ex. 1004 at 80. Indeed, the CompactFlash signals are communicated to the host using an interrupt pin to alert the host to the change of state. *Id.* at 24-26, 29, Tables 4-1, 4-2.

A POSA would have understood that the two busy signals of MMC 3.31 could be implemented in either the open-ended or the pre-defined multiple block write operation. *Supra*, X.A.5 (Claim 19[d]); Ex. 1015 at ¶178. As described above in Section IX.A, MMC 3.31 discloses that the card uses busy signals during an open-ended multiple block write operation. The busy signal sent between blocks indicates that the buffers are full. After the stop command tells the card that the last block has been received, the busy signal indicates that the card is programming the received data blocks. Ex. 1003 at 35, Figs. 10, 30-34; Ex. 1015 at ¶178. As described above, MMC 3.31 also describes a multiple block write command that uses a pre-defined block count and omits the stop command. Ex. 1003 at 38. Other than omitting the stop command, the functionality of the card is unchanged whether the open-ended multiple block write or pre-defined multiple block write is used. Ex. 1003 34-35; *see supra*, X.A.5 (Claim 19[d]).

Thus, a POSA would have been motivated to apply the same busy signaling technique used in the open-ended multiple block write for the pre-defined multiple block write. Ex. 1015 at ¶178. A POSA would realize that the busy signaling technique that was implemented with respect to an open-ended multiple block write operation would improve the pre-defined multiple block write in the same way. Ex. 1015 at ¶178; *KSR Intern. Co.*, 550 U.S. 398 at 417.

MMC 3.31 discloses that a transfer may end without a stop command. Ex. 1003 at 38. Figure 19 of MMC 3.31 shows that data transfer may end either when a stop command is received or when the transfer ends. Ex. 1003 at 34, Fig. 19. A POSA would have found it obvious that the transfer would end when the card is aware that all data blocks have been transferred. Ex. 1015 at ¶179. The card knows that this is a "transfer end" situation because the block count was set by CMD 23 prior to the multiple block write operation being initiated. Ex. 1015 at ¶179.

MMC 3.31 also discloses that the busy signal indicates both that the buffer is busy and that the card is busy programming. *Supra*, X.5.A (Claim 19[d]). A POSA would have found it obvious that the busy signal could have both meanings based on the description of the state diagram in MMC 3.31. Ex. 1003 at 34-35, Fig. 19; *Supra*, X.A.5 (Claim 19[d]). MMC 3.31 discloses that "[i]f all write buffers are full, and as long as the card is in *Programming State* (see . . . Figure 19), the DAT line will be kept low." Ex. 1003 at 35. This passage shows that the busy signal could indicate buffer status in some scenarios while indicating programming in others. Ex. 1015 at ¶180. The MMC 3.31 specification confirms that a card may use a busy signal to indicate programming regardless of buffer status. Ex. 1015 at ¶ 180. Figure 34 of MMC 3.31 shows the use of the busy

signal to indicate programming when the card's buffers are not full. Ex. 1015 at ¶180.

In addition, MMC 3.31 discloses communicating a programming status in connection with the Serial Peripheral Interface mode. Ex. 1003 at 94; Ex. 1010 at 11:14-19; Ex. 1015 at ¶180. Based on these disclosures, a POSA would have understood that the card uses the busy signal to have two distinct meanings. Ex. 1015 at ¶180.

As explained above, the express disclosure of a busy signal with two different meanings in the open-ended multiple block write operation and the description of the state diagram for both multiple block write operations would have provided an express motivation to a POSA to apply this signaling scheme to the pre-defined multiple block write operation. Ex. 1015 at ¶181.

Similarly, a POSA would have been expressly motivated to apply a busy signal with two different meanings to the pre-defined multiple block write operation based on the teachings of CompactFlash in order to further achieve the efficiency goals of MMC 3.31. Ex. 1003 at 11; Ex. 1015 at ¶181. Based on the knowledge that CompactFlash uses an interrupt to indicate that programming has been completed following a data transfer, a POSA would have found it obvious to use the MMC 3.31 busy signaling, which was already being used during open-

ended multiple block write operations, to indicate that programming had been completed following a pre-defined multiple block write. Ex. 1015 at ¶182. Because one of the stated goals of both MMC 3.31 and CompactFlash is high throughput, a POSA looking to improve upon MMC 3.31 would be motivated to look to CompactFlash, another standard for high-capacity, portable, low weight memory cards intended to act as mass storage devices. *Id.* Indeed, CompactFlash supports the ATA standard, the standard used for disk drives for decades, and is therefore a logical source for improvements to MMC 3.31. *Id*.

One way CompactFlash achieves high throughput is the use of interrupt signals by which the card tells the host when its buffer is ready and when programming is complete. Ex. 1004 at 7, 80. These interrupt signals avoid the need to poll the card to ascertain the card's status. *Id.*; Ex. 1015 at ¶182. The write sector(s) command of CompactFlash is similar to the pre-defined multiple block write of MMC 3.31 because both are commands to write a pre-defined number of data blocks/sectors to a portable flash memory device. Ex. 1015 at ¶182.

Based on CompactFlash disclosing this signaling to achieve high performance in a multiple sector write operation, and because the purpose of MMC 3.31's pre-defined multiple block write operation is to achieve high performance

when writing multiple blocks to a removable flash memory device, a POSA would have found it obvious to send a busy signal indicating when a memory card is programming after the last block of a pre-defined multiple block write operation is received. Ex. 1015 at ¶ 183. With such a modification, which is already suggested by the teachings of MMC 3.31 itself as discussed above, an MMC card would send busy signals having two different meanings—buffer full and programming—within the same command execution, like the interrupts of CompactFlash. Ex. 1015 at ¶183. Similar to the busy signal of MMC 3.31, the interrupts of CompactFlash indicate that the buffers are busy for all but the last sector, and indicate that the card is programming following the last sector. Ex. 1015 at ¶183. The fact that MMC 3.31 discloses detailed descriptions of how busy signals are used during an open-ended multiple block write operation provides further motivation to combine the techniques, since the busy signals of MMC 3.31 in these circumstances and the interrupt technique of CompactFlash are very similar. Ex. 1015 at ¶183.

A POSA would have been aware of the known technique of using interrupts to allow a CompactFlash card to indicate its status to a host device during a data transfer. Ex. 1015 at ¶184. A POSA would also have understood that a technique implemented in a CompactFlash memory card could improve an MMC memory

card, a similar device, in the same way. *KSR Intern. Co.*, 550 U.S. 398 at 417 ("[I]f a technique has been used to improve one device, and a POSA would recognize that it would improve similar devices in the same way, using the technique is obvious unless its actual application is beyond his or her skill."); Ex. 1015 at ¶184. As discussed above, a POSA would find it obvious, based on the knowledge of how CompactFlash uses interrupts, to implement busy signaling within the multiple block write operation to indicate to the host that the card is programming following the last block of the data transfer. Ex. 1015 at ¶184.

A. Claim 19

1. Claim 19 [preamble]

To the extent that the preamble is limiting, MMC 3.31 discloses "[a] memory device." *Supra*, X.A.1 (Claim 19 [preamble]).

2. Claim 19[a]

MMC 3.31 discloses "a bus interface configured to be coupled to a host through a bus having a data signal line." *Supra*, X.A.2 (Claim 19[a]).

3. Claim 19[b]

MMC 3.31 discloses "the bus interface further comprising a driver at said memory device coupled to said data signal line and a receiver at said memory device coupled to the data signal line, said receiver being operable to receive information comprising a first information portion and a second information

portion from the host over the data signal line within a command execution." *Supra*, X.A.3 (Claim19[b]).

4. Claim 19[c]

MMC 3.31 discloses "said driver being operable to drive a change of state of the data signal line to the host within the command execution." *Supra*, X.A.4 (Claim 19[c]).

5. Claim 19[d]

MMC 3.31 renders obvious this element: "said bus interface further comprising a controller coupled to said driver and to said receiver and operable to cause the change of state of the data signal line to have a first meaning after receiving the first information portion within the command execution and to have a second meaning different from the first meaning after receiving the second information portion within the command execution from the host over the data signal line." *Supra*, X.4.5 (Claim 19[d]).

To the extent that MMC 3.31 alone is not deemed to render causing the change of the data signal line "to have a second meaning different from the first meaning after receiving the second information portion within the command execution from the host over the data signal line" obvious, CompactFlash discloses this limitation.

Similar to the commands disclosed by MMC 3.31, CompactFlash discloses commands for transferring multiple portions, e.g. sectors, of data to a card using a command. Ex. 1004 at 80; Ex. 1015 at ¶191. For example, CompactFlash discloses a Write Sector(s) command that "writes from 1 to 256 sectors" of data. Ex. 1004 at 80. These sectors are similar to the blocks of MMC 3.31. Ex. 1015 at ¶191. The Write Sector(s) command, like the open-ended multiple block write command of MMC 3.31, signals to the host when the card's buffer is ready to receive additional data by driving a change of state on an interrupt pin. Ex. 1004 at 80; Ex. 1015 at ¶191.

After receiving each sector, the card sets a busy bit in the Status Register. Ex. 1004 at 53, 55, 80. When the card buffer is ready to receive the next sector, the busy bit is cleared and the interrupt signal is generated. *Id*. This interrupt is a change of state that signals to the host that the buffer can receive additional data. *Id*.; Ex. 1015 at ¶192.

Execution of the write sectors command does not require a stop command to stop the data transfer, but writes the number of "sectors as specified in the Sector Count Register." Ex. 1004 at 80; Ex. 1015 at ¶197. After the final sector is transferred, a busy bit is again set in the Status Register. *Id.* When the command is completed (i.e., the data has been programmed), the busy bit is cleared and the

card generates the same interrupt. *Id.*; Ex. 1015 at ¶¶193, 197. This final interrupt signal, like the interrupts between sectors, is a change of state that communicates status to the host. Ex. 1015 at ¶¶193, 197. In this case, the meaning of the change of state is that programming is complete. Ex. 1004 at 80; Ex. 1015 at ¶¶193, 197.

A POSA would have been motivated to apply a busy signal with two different meanings to the pre-defined multiple block write operation based on the above-noted teachings of CompactFlash in order to further achieve the efficiency goals of MMC 3.31. Ex. 1003 at 11; Ex. 1015 at ¶194. Based on the knowledge that CompactFlash uses an interrupt to indicate that programming has been completed following a data transfer, a POSA would have found it obvious to use the MMC 3.31 busy signaling, which was already being used during open-ended multiple block write operations, to indicate that programming had been completed following a pre-defined multiple block write. Ex. 1015 at ¶194. Because one of the stated goals of both MMC 3.31 and CompactFlash is high throughput, a POSA looking to improve upon MMC 3.31 would be motivated to look to CompactFlash, another standard for high-capacity, portable, low weight memory cards intended to act as mass storage devices. *Id.* One way CompactFlash achieves high throughput is the use of interrupt signals by which the card tells the host when its buffer is ready and when programming is complete. Ex. 1004 at 7, 80. These interrupt

signals avoid the need to poll the card to ascertain the card's status. *Id.*; Ex. 1015 at ¶194. The write sector(s) command of CompactFlash is similar to the pre-defined multiple block write of MMC 3.31 because both are commands to write a pre-defined number of data blocks/sectors to a portable flash memory device. Ex. 1015 at ¶194.

Based on CompactFlash's disclosed signaling to achieve high performance in a multiple sector write operation, and because the purpose of MMC 3.31's pre-defined multiple block write operation is to achieve high performance when writing multiple blocks to a removable flash memory device, a POSA would have found it obvious to send a busy signal indicating when a memory card is programming after the last block (i.e., the claimed "second information portion") of a pre-defined multiple block write operation is received in MMC 3.31. Ex. 1015 at ¶195. With such a modification, which is already suggested by the teachings of MMC 3.31 itself as discussed above, an MMC card would send busy signals having two different meanings—buffer full and programming—within the same command execution, like the interrupts of CompactFlash. Ex. 1015 at ¶195. Similar to the busy signal of MMC 3.31, the interrupts of CompactFlash indicate that the buffers are busy for all but the last sector, and indicate that the card is programming following the last sector. Ex. 1015 at ¶195.

A POSA would have been aware of the known technique of using interrupts to allow a CompactFlash card to indicate its status to a host device during a data transfer. Ex. 1015 at ¶195. A POSA would also have understood that a technique implemented in a CompactFlash memory card could improve an MMC memory card, a similar device, in the same way. *KSR Intern. Co.*, 550 U.S. 398 at 417 ("[I]f a technique has been used to improve one device, and a POSA would recognize that it would improve similar devices in the same way, using the technique is obvious unless its actual application is beyond his or her skill."); Ex. 1015 at ¶195.

Further, a POSA would have found it obvious that a mechanism for the memory card to communicate with the host in CompactFlash would increase the efficiency of other memory cards using write commands of multiple units of transfer, such as the memory card specified in MMC 3.31. Ex. 1015 at ¶196. A POSA would have found it obvious that the busy signaling already disclosed by MMC 3.31 could be used to achieve this efficiency. Ex. 1015 at ¶196.

As discussed above, combining the interrupt signals of CompactFlash's write sectors command with the busy signaling of MMC 3.31's pre-defined multiple block write operation produces a system with two busy signals (with different meanings) asserted within the same command, i.e. within CMD 25 of MMC 3.31. In such a case, the card would send a busy signal indicating the buffer

is full once a first data block is received and indicating the card is programming once the last data block is received, which the card would be able to identify based on the number of blocks to be written that was defined (via CMD23) before the multiple block write command was issued. Ex. 1015 at ¶196.

A POSA therefore would have found it obvious to incorporate CompactFlash's Write Sector(s) command into MMC 3.31 so that MMC 3.31's multiple block write command did not require a stop command (CMD12), but set the number of blocks to be transferred in advance of the multiple block write command. Ex. 1015 at ¶198. Indeed, MMC 3.31 disclosed this technique with regard to the set block count command, but did not disclose how the pre-defined multiple block write would operate with regard to busy signaling. However, a POSA also would have found it obvious, based on the knowledge that CompactFlash uses an interrupt to signal a change of state to the host, to modify the busy signaling of MMC 3.31 so that MMC 3.31's multiple block write command would use busy signaling to communicate that a buffer is full after a first data block is received and to communicate the card is programming after the last block is received. Ex. 1015 at ¶198. Once this technique of CompactFlash is combined with MMC 3.31, the busy signals of MMC 3.31 occur "within the

command execution" (i.e., during execution of the pre-defined multiple block write command). Ex. 1015 at ¶198.

Accordingly, to the extent that MMC 3.31 is not deemed to render causing the change of the data signal line "to have a second meaning different from the first meaning after receiving the second information portion within the command execution from the host over the data signal line" obvious, CompactFlash discloses this limitation, and a POSA would have been motivated to apply busy signaling with two different meanings to the pre-defined multiple block write operation in MMC 3.31 within a same command execution based on the teachings of CompactFlash. Ex. 1015 at ¶199.

In view of the foregoing, MMC 3.31 and CompactFlash render obvious claim element 19[d]. Ex. 1015 at ¶200.

B. Claim 20

Claim 20 depends from Claim 19, and requires that the bus is comprised of a command signal line and that the controller is responsive to at least one command for a multi-block transfer that initiates the command execution received through the command signal line from the host. MMC 3.31 discloses that all data transfer commands are sent over the CMD line, which is separate from the DAT line. Ex.

1003 at 18, 21-22. The multiple block write command is sent over the CMD line and causes multiple blocks to be written to the memory card. Ex. 1003 at Fig. 10.

As noted above, MMC 3.31 discloses that the memory card contains a controller coupled to an interface driver. Ex. 1003 at 20, Fig. 4. A POSA would understand that this controller is part of the bus interface and implements the communication protocol disclosed by MMC 3.31, including how the host should interpret command signals from the memory card. Ex. 1003 at 14; Ex. 1015 at ¶201.

Further, as discussed with respect to Claim 19, MMC 3.31 and CompactFlash disclose that when executing the multiple block write command, the change of state of the data signal line is caused to have a first meaning after receiving the first data block, and to have a second meaning after receiving the second data block. *See supra* Section XI.A.5, Claim 19, e.g., Claim 19[d]; *see also* Ex. 1015 at ¶202.

Accordingly, MMC 3.31 and CompactFlash render obvious Claim 20. Ex. 1015 at ¶203.

XII. CONCLUSION

For the foregoing reasons, Petitioner respectfully requests that a trial for *inter partes* review of the '469 Patent be instituted and that Claims 19 and 20 be rejected and canceled.

Dated: January 31, 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)

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

Attorneys for Petitioner
Kingston Technology Company, Inc.

CERTIFICATE OF COMPLIANCE

1. The undersigned certifies that this brief complies with the type volume limitations of 37 CFR § 42.24(a)(1)(i). This brief contains 13,456 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 2016, the word processing program used to create it.

2. 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 2016 in Times New Roman 14-point font

Dated: January 31, 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)

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

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. 7,565,469** and supporting materials (Exhibits 1001 - 1018 and Power of Attorney) have been served in its entirety this 31st of January, 2019, by e-mail and Federal Express on Patent Owner at the correspondence address for the attorney of record for the 7,565,469 Patent shown in USPTO PAIR, as well as on counsel for Patent Owner in the co-pending litigation:

Patent Owner in PAIR (*via FedEx*):

Daniel Hayes
LEE & HAYES, P.C.
601 W. Riverside Avenue, Suite 1400
Spokane, WA 99201

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)