(19) **Bundesrepublik Deutschland**
Deutsches Patent- und Markenamt

(10) **DE 197 33 151 B4** 2005.12.08

(12) # Patentschrift

(21) Aktenzeichen: **197 33 151.3**
(22) Anmeldetag: **31.07.1997**
(43) Offenlegungstag: **09.07.1998**
(45) Veröffentlichungstag
der Patenterteilung: **08.12.2005**

(51) Int Cl.[7]: **G06F 13/362**
G06F 13/10

(30) Unionspriorität:
08/774,452 30.12.1996 US

(73) Patentinhaber:
Hewlett-Packard Development Co., L.P., Houston, Tex., US

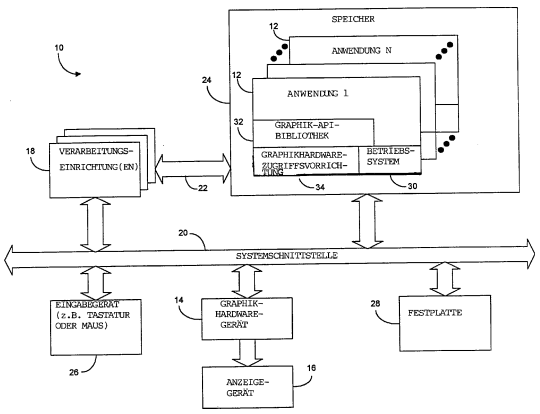
(74) Vertreter:
Schoppe, F., Dipl.-Ing.Univ., Pat.-Anw., 82049 Pullach

(72) Erfinder:
Blaho, Bruce, Fort Collins, Col., US;
Goeltzenleuchter, Courtney, Fort Collins, Col., US;
Herrell, Russ W., Fort Collins, Col., US

(56) Für die Beurteilung der Patentfähigkeit in Betracht gezogene Druckschriften:
US 55 06 969 A
US 53 27 540 A

(54) Bezeichnung: **Vorrichtung und Verfahren für einen virtuellen Gerätezugriff in einem Computersystem**

(57) Hauptanspruch: Vorrichtung für einen virtuellen Gerätezugriff zum Vorsehen einer Verriegelung ohne Zusatzaufwand eines Graphikhardwaregeräts (14) eines Computersystems (10) für Anwendungen (12), die auf dem Computersystem (10) laufen, mit folgenden Merkmalen:
einer Hardwaresteuerungsverwaltungseinrichtung (40) zum Erfassen, wann eine der Anwendungen (12) versucht, auf das Graphikhardwaregerät (14) zuzugreifen, und zum Freigeben und Sperren des Zugriffs auf das Graphikhardwaregerät (14);
einer Anwenderebene-Signalhandhabungseinrichtung (42), die jeder der Anwendungen (12) zugeordnet ist, wobei jede der Anwenderebene-Signalhandhabungseinrichtungen (42) zum Durchführen einer Graphikkontextumschaltung konfiguriert ist, wobei, wenn eine der Anwendungen (12) versucht, auf das Graphikhardwaregerät (14) zuzugreifen, die Hardwaresteuerungsverwaltungseinrichtung (40) ein Signal an die Anwenderebene-Signalhandhabungseinrichtung (42) ausgibt, die der Anwendung zugeordnet ist, die versucht, auf das Graphikhardwaregerät zuzugreifen, wobei die Anwenderebene-Signalhandhabungseinrichtung, die das Signal von der Hardwaresteuerungsverwaltungseinrichtung empfangen hat, eine Graphikkontextumschaltung als Reaktion auf den Empfang des Signals von der Hardwaresteuerungsverwaltungseinrichtung durchführt; und
einem Shared-Memory-Betriebsmittel (44), auf das durch jede der Anwendungen (12) zugegriffen werden...



Beschreibung

[0001] Die vorliegende Erfindung bezieht sich allgemein auf einen Steuerungs- und Verwaltungszugriff auf eine Graphikhardware in einem Computersystem und insbesondere auf eine Vorrichtung und ein Verfahren für einen virtuellen Zugriff einer Graphikhardware in einem Computersystem ohne Leistungseinbußen pro Transaktion.

Stand der Technik

[0002] Ein Graphikhardwaregerät wird gewöhnlicherweise in einem Computersystem verwendet, um zwei- oder dreidimensionale Darstellungen eines Objekts auf einem zweidimensionalen Bildschirm eines Anzeigegeräts aufzubereiten. Typischerweise ist ein auf dem Anzeigebildschirm aufzubereitendes Objekt in eine Mehrzahl von Graphikgrundelementen unterteilt. Die Graphikgrundelemente sind Grundkomponenten eines Graphikbildes und können durch die Geometrie eines Punktes, einer Linie, eines Vektors oder eines Polygons, wie z. B. eines Dreiecks, definiert werden. Um die Grundelemente eines Objekts aufzubereiten, werden die Grundelemente durch eine Graphikpipeline geführt, in der verschiedene Verarbeitungstypen auftreten, beispielsweise eine Transformation, eine Beleuchtung für eine Schattenerzeugung, ein Abschneiden, eine perspektivische Unterteilung und eine Abtastungsumwandlung. Die Operationen einer Graphikpipeline werden typischerweise von der Graphikhardware durchgeführt, welche die Grundelementdaten von einer Anwendung empfängt, die auf dem Computersystem läuft.

[0003] Aufgrund der Konkurrenz auf dem Computersoftwaremarkt ist die Zeit, die eine Anwendung in Anspruch nimmt, um ein Objekt auf einem Anzeigegerät aufzubereiten, von besonderer Bedeutung. Die erforderliche Zeitdauer wird in hohem Maße von der Zeitdauer bestimmt, die jeder Anwendungsschnittstellenaufruf ("API-call"; API-call = Application Interface Call = Anwendungsschnittstellenaufruf) für eine Ausführung benötigt. Beispielsweise erfordern die Graphikbeschleuniger, welche ihre eigene Geometrieverarbeitung durchführen, daß die Graphik-API die Anwendungsbilddaten direkt an die Graphikhardware schickt. Ferner muß ein gegebener Graphik-API-Eintrittspunkt sicherstellen, daß sein Graphikkontext gegenwärtig in die Hardware geladen wird, bevor dieselbe eine Übergabe einleiten kann, um zu verhindern, daß sich mehrere Graphikprozesse gegeneinander stören. Diese Überprüfung erhöht den Zusatzaufwand des Graphik-API-Aufrufs beträchtlich. Bei Hochleistungssystemen verarbeiten dedizierte Graphikbeschleuniger die Graphikdaten. Folglich müssen die Graphik-APIs die Daten an die Graphikhardware mit einer minimalen Menge an Zusatzaufwand übergeben, um eine maximale Leistungsfähigkeit zu erreichen. Folglich kann der Zusatzaufwand, der dem Zustandebringen einer Steuerung über die Graphikhardware zugeordnet ist, die Geschwindigkeitsvorteile des Graphikbeschleunigers effektiv beseitigen.

[0004] Von besonderer Bedeutung sind Scheitelpunkt-basierte APIs, bei denen jede Komponente eines Grundelements (d. h. Farb-, Normal-, Texturkoordinate oder -Scheitelpunkt) mit einem separaten Bibliotheksaufruf spezifiziert ist, wobei jeder einen Hardwarezugriff erfordert. Daher beläuft sich mit Scheitelpunkt-basierten APIs die Zeit, die erforderlich ist, um die Graphikhardware zu verriegeln (d. h. einen ausschließlichen Zugriff bereitzustellen) und zu entriegeln auf einen viel größeren Anteil an der Arbeit, die beim Aufbereiten des Grundelements durchgeführt wird, als es mit Zeichenelementbasierten APIs der Fall ist, bei denen lediglich ein Bibliotheksaufruf pro Grundelement durchgeführt wird. Folglich verringert der Zusatzaufwand, der dem Verriegeln und Entriegeln der Graphikhardware für jede Komponente eines Zeichenelements zugeordnet ist, deutlich das Verhalten von Scheitelpunkt-basierten APIs.

[0005] Sobald die API einer Anwendung auf die Graphikhardware zugegriffen hat, beginnt die API deren Daten zu der Graphik-Hardware zu übertragen. Die Zeit, die von der Graphikhardware benötigt wird, um Daten zu verarbeiten, kann abhängig von den Daten und den Prozessen, die von dem Graphikbeschleuniger durchgeführt werden, um mehrere Größenordnungen variieren. Folglich ist es möglich, daß die Datenpuffer der Graphikhardware voll werden und Daten verloren gehen können, da die Graphikhardware keine weiteren Eingangsdaten akzeptieren kann. Folglich ist eine bestimmte Menge an Zusatzaufwand in der Graphik-API auf ein Verwalten des Datenflusses zurückzuführen, um zu überprüfen, daß die Datenpuffer der Graphikhardware genügend Raum für die Daten haben, die gerade von einer Anwendung zu der Graphikhardware geschickt werden.

[0006] Demgemäß besteht in der Industrie ein Bedarf nach einem System und einem Verfahren zum Zugreifen auf die Graphikhardware eines Computersystems ohne Zusatzaufwand pro Transaktion. Dies ist bezüglich Scheitelpunkt-basierter APIs, bei denen jeder Bibliotheksaufruf einen Hardwarezugriff ergibt, ein spezieller Bedarf. Außerdem wäre es wünschenswert, den Datenfluß von der Graphik-API zu der der Graphikhardware ohne gerätspezifische Kernunterstützung von dem Betriebssystem und mit einer minimalen Menge an Zusatzaufwand in der API zu steuern, derart, daß keine Daten verloren werden, da das Graphikhardwaregerät die Daten schnell genug verarbeiten kann.

[0007] Die US 5,327,540 A beschäftigt sich mit einer Computerarchitektur und ist auf ein Verfahren und eine Vorrichtung gerichtet, um zu ermöglichen, daß

verschiedene Mikrokanal-Bus-Master mit einem Eingangs/Ausgangs-Bus verbunden werden, um auf einen Systembus zuzugreifen, so daß Daten aus einem gemeinsam verwendeten Speicherbetriebsmittel, das mit dem Systembus verbunden ist, gelesen werden können bzw. in dieses geschrieben werden können. Der Eingabe/Ausgabe-Bus ist mit einem Mikrokanal-Schnittstellenmodul verbunden, der den Zugriff auf die Speichermodule, die mit den Systembussen verbunden sind, durch die Mikrokanal-Bus-Master steuert. Das Mikrokanal-Schnittstellenmodul umfaßt einen Arbiter, der Entscheidungsaufgaben durchführt, um zu bestimmen, welcher der Mikrokanal-Bus-Master auf die mit dem Systembus verbundenen Speichermodule zugreifen kann.

[0008] Jeder der Mikrokanal-Bus-Master weist eine 4-Bit-Entscheidungsebene auf, die diesem zugeordnet ist. Jede der Entscheidungsebenen entspricht einem bestimmten Prioritätswert. Wenn einer der Mikrokanal-Bus-Master versucht, auf den Systembus zuzugreifen, wird die Entscheidungsebene, die dem bestimmten Mikrokanal-Bus-Master zugeordnet ist, durch das Mikrokanal-Schnittstellenmodul beurteilt. Die 4-Bit-Entscheidungsebene wird durch das Mikrokanal-Schnittstellenmodul in einen Prioritätswert decodiert. Wenn mehrere Mikrokanal-Bus-Master versuchen, auf den Systembus zuzugreifen, werden die Bus-Master bestimmen, ob einer der anderen Bus-Master einen höheren Prioritätswert aufweist oder nicht, und derjenige Bus-Master, der nicht den höchsten Prioritätswert hat, wird ein Treiben des Eingabe/Ausgabebusses unterbrechen. Der Bus-Master mit dem höchsten Prioritätspegel fährt fort, den Eingabe/Ausgabe-Bus zu treiben, und das Mikrokanal-Schnittstellenmodul verwendet den Prioritätswert dieses Bus-Masters, um einen Moduswert zu erhalten. Dieser Moduswert spezifiziert die Art, auf die der Bus-Master Daten zwischen dem mit dem Systembus verbundenen Speichermodul und dem Bus-Master, der den Eingabe/Ausgabe-Bus treibt, übertragen wird.

[0009] Die US-5,506,969 ist darauf gerichtet, gemeinsam verwendete Computerressourcen effizient zu nutzen, und befasst sich insbesondere mit einem Hochgeschwindigkeitsbus. Wenn eine Übertragungsanfrage durch eine Anwendung erzeugt wird, wird diese in ein globales Steuerungsbefehlspaket, ein Zielmodulsteuerungspaket und ein Quellmodulsteuerungspaket zerteilt, die dann verwendet werden, um die Daten effektiv über den Hochgeschwindigkeitsbus zu den geeigneten Speicherorten zu übertragen. Es wird jedoch keine Graphikkontextumschaltung durchgeführt.

Aufgabenstellung

[0010] Die Aufgabe der vorliegenden Erfindung besteht darin, eine Vorrichtung und ein Verfahren zu schaffen, um ohne zusätzlichen Mehraufwand pro Transaktion auf die Graphikhardware eines Computersystems zugreifen zu können.

[0011] Die Aufgabe der vorliegenden Erfindung wird durch eine Vorrichtung für einen virtuellen Gerätezugriff gemäß Patentanspruch 1 und durch ein Verfahren für einen virtuellen Gerätezugriff gemäß Anspruch 11 gelöst.

[0012] Die vorliegende Erfindung überwindet die Unzulänglichkeiten und Defizite des Stands der Technik, die im vorhergehenden hierin erörtert wurden und in der Industrie bekannt sind. Die vorliegende Erfindung sorgt für einen virtuellen Zugriff der Graphikhardware in einem Computersystem durch eine oder mehrere Anwendungen, die asynchron auf dem Computersystem laufen. Folglich können die Anwendungen ohne eine explizite Überprüfung oder einem Protokoll in der Graphikanwendungsprogrammschnittstelle (API) auf das Graphikhardwaregerät zugreifen, als ob sie einen ausschließlichen Zugriff hätten.

[0013] Gemäß der vorliegenden Erfindung ist ein Zugriff des Graphikhardwaregeräts in eine gerätunabhängige Steuerung, die sich in dem Betriebssystem befindet, und eine gerätabhängige Steuerung unterteilt, die sich in der Graphik-API jeder der Anwendungen befindet, die auf dem Computersystem arbeiten. Die gerätunabhängige Steuerung wird von einer Hardwaresteuerungsverwaltungseinrichtung durchgeführt, die in dem Betriebssystem vorgesehen ist und die im wesentlichen steuert, wer auf die Graphikhardware und wann darauf Zugriff hat. Die gerätabhängige Steuerung wird von einer Signalhandhabungseinrichtung durchgeführt, die in der Graphik-API jeder Anwendung vorgesehen ist und die im wesentlichen das Umschalten des Zugriffs auf die Graphikhardware zwischen den Anwendungen steuert. Zusätzlich ist ein Shared-Memory-Bereich (shared memory = gemeinsam verwendeter Speicher) vorgesehen, auf den von jeder der Anwendungen zugegriffen werden kann, und der zum Speichern der Identität der Anwendung, die gegenwärtig auf die Graphikhardware zugreift, und des Graphikhardwarezustands jeder Anwendung konfiguriert ist.

[0014] Die Hardwaresteuerungsverwaltungseinrichtung umfaßt eine Steuerlogik, die es jeder Anwendung ermöglichen wird, bei einer Anforderung auf die Graphikhardware zuzugreifen, jedoch lediglich nachdem der Graphikkontextzustand dieser Anwendung in die Graphikhardware geladen worden ist. Die Hardwaresteuerungsverwaltungseinrichtung verwendet Zugriffsschutzidentifizierer, um zu erfassen, wann eine Anwendung wünscht, auf die Graphikhardware zuzugreifen. Wenn ein Schutzidentifiziererfehler erfaßt ist, schickt die Hardwaresteuerungsverwaltungseinrichtung ein Signal zu der Anwendung,

die einen Zugriff versucht. Das Signal leitet eine Signalabwicklung auf der Anwenderebene ein, die den bestehenden Graphikzustand sichern und den Graphikzustand für diese Anwendung wieder herstellen muß. Sobald der Graphikhardwarezustand der Anwendung in die Graphikhardware geladen worden ist, wird dieser Anwendung von der Hardwaresteuerungsverwaltungseinrichtung die Erlaubnis gegeben, auf die Graphikhardware zuzugreifen. Sobald der Fehler gelöst worden ist, kann die Anwendung folglich auf das Graphikhardwaregerät ohne Zusatzaufwand zugreifen, als ob dieselbe die einzige Anwendung wäre, die mit der Graphikhardware spricht.

[0015] Die Signalhandhabungseinrichtungen werden auf der Anwenderebene als Teil der Graphik-API-Bibliothek jeder Anwendung implementiert, wobei dieselben folglich eine gerätabhängige Steuerung, wie z. B. ein Graphikkontextumschalten, wirksam handhaben können. Bei dem bevorzugten Ausführungsbeispiel ist die Signalhandhabungseinrichtung der Anwendung, die einen Zugriff auf die Graphikhardware versucht, für das Durchführen der Graphikkontextumschaltungen verantwortlich. Dies umfaßt ein Sichern des Graphikhardwarezustandes, der der Anwendung zugeordnet ist, die gegenwärtig auf die Graphikhardware zugreift, und ein Wiederherstellen des Graphikhardwarezustands der Anwendung, die versucht, auf die Graphikhardware zuzugreifen. Demgemäß wird das Umschalten des Graphikhardwarezustands in der Graphikhardware zwischen den Anwendungen im Hintergrund durchgeführt, derart, daß die Anwendungen nicht erkennen, daß eine weitere Anwendung entweder einen Zugriff der Graphikhardware übernimmt oder einen Zugriff der Graphikhardware von einer anderen Anwendung anfordert. Folglich ist jede Anwendung mit einem virtuellen Gerätezugriff versehen.

[0016] Die vorliegende Erfindung sorgt ferner wie folgt für einen virtuellen Zugriff auf die Graphikhardware eines Computersystems und kann als Verfahren für denselben konzipiert sein. Zu Anfang versucht eine Anwendung, die auf einem Computersystem läuft, auf die Graphikhardware zuzugreifen. Dies verursacht einen Schutzidentifiziererfehler, der von der Hardwaresteuerungsverwaltungseinrichtung als ein Versuch erkannt wird, die Graphikhardware zu verwenden. Zusätzlich wird der Anwendungsprozeß blockiert, sowie die Hardwaresteuerungsverwaltungseinrichtung den Fehler verarbeitet. Aus der Perspektive der Anwendung hat die Anwendung jedoch lediglich einen gewöhnlichen Graphikzugriff durchgeführt. Es besteht die Vorstellung, daß dieselbe die einzige Anwendung ist, die einen Zugriff auf die Graphik-Hardware hat. Falls die Graphikhardware verfügbar ist, wird die Hardwaresteuerungsverwaltungseinrichtung den Anwendungszugriff auf die Graphikhardware sofort gewähren. Anderenfalls bleibt die Anwendung blockiert, wobei die Hardwaresteuerungsverwaltungseinrichtung warten muß, bis die Anwendung, die gegenwärtig auf die Graphikhardware zugreift, unterbrochen werden kann.

[0017] Falls jedoch die gegenwärtige Anwendung unterbrochen werden kann, schickt die Hardwaresteüerungsverwaltungseinrichtung ein Steuersignal zu der Signalhandhabungseinrichtung der Anwendung, die einen Zugriff versucht, wodurch die Signalhandhabungseinrichtung angewiesen wird, eine Graphikkontextumschaltung durchzuführen. Wie es im vorhergehenden erwähnt wurde, umfaßt eine Graphikkontextumschaltung ein Sichern des Graphikhardwarezustands der Anwendung, die gegenwärtig auf das Graphikhardwaregerät zugreift, in dem gemeinsam verwendeten Speicher und ein Wiederherstellen des Graphikhardwarezustands der Anwendung, die versucht, auf das Graphikhardwaregerät zuzugreifen. Wenn die Signalhandhabungseinrichtung die Graphikkontextumschaltung abgeschlossen hat, benachrichtigt die Signalhandhabungseinrichtung die Hardwaresteuerungsverwaltungseinrichtung. Dieser Prozeß ist beendet, wenn die Hardwaresteuerungsverwaltungseinrichtung den Anwendungszugriff auf das Gerät gewährt.

[0018] Jeder nachfolgende Versuch durch eine Anwendung, auf die Graphikhardware zuzugreifen, wird im wesentlichen auf dieselbe Art und Weise, wie es im vorhergehenden beschrieben wurde, abgewickelt. Dennoch sei angemerkt, daß jede Anwendung, der die Erlaubnis gegeben ist, auf das Graphikhardwaregerät zuzugreifen, zumindest eine minimale Graphikzeitscheibe zugestanden wird, um eine unannehmbares Aufeinandertreffen zwischen konkurrierenden, parallel-laufenden Anwendungen zu verhindern.

[0019] Wie es im vorhergehenden in dem Hintergrund-Abschnitt beschrieben wurde, gibt es außerdem Zeiten, zu denen die Graphikhardware keine Daten mehr annehmen kann. In diesen Fällen muß der Fluß der Graphikdaten eingedämmt werden, jedoch auf eine Weise, die einen geringen oder keinen Zusatzaufwand für die Graphik-API-Bibliothek der Anwendung verursacht. Demgemäß sorgt die vorliegende Erfindung für eine Flußsteuerungsvorrichtung, um dieses Ziel zu erreichen, indem zuerst der Datenfluß verlangsamt wird, wenn die Eingangsdatenpuffer der Graphikhardware einen ersten vorbestimmten Punkt erreichen, und falls nötig der Datenfluß angehalten wird, wenn die Eingangsdatenpuffer einen zweiten vorbestimmten Punkt erreichen.

[0020] Die Flußsteuerungsvorrichtung verlangsamt den Datenfluß wesentlich, wenn als erstes erfaßt wird, daß die Graphikhardware zurückbleibt. Eine niedrige Füllstandsmarke in dem Eingangsdatenpuffer der Hardware löst aus, daß die Schnittstellensteuerlogik damit beginnt, langsamer auf die Bustransaktionen zu antworten. Diese Verzögerungen werden

sich eventuell zurück zu der Verarbeitungseinrichtung ausbreiten, die die Daten schreibt, wodurch die Daten ohne zusätzlichen Zusatzaufwand in der Graphik-API gedrosselt werden.

[0021] Falls die langsamere Datenrate für die Graphikhardware noch zu schnell ist, um Schritt zu halten, und der Eingangsdatenpuffer sich weiter füllt, löst eine hohe Füllstandsmarke in den Eingangsdatenpuffern eine gerichtete Unterbrechung (Interrupt) aus, die direkt zu der Verarbeitungseinrichtung geschickt wird, die die Daten schreibt. Das Betriebssystem wird die Unterbrechung erfassen und erkennen, daß die Graphikhardware einen Rückstau aufweist, und die Erlaubnis entfernen, um von einer beliebigen Anwendung auf die Graphikhardware zuzugreifen. Bei einem Mehrprogrammverarbeitungssystem ist es kritisch, daß die Unterbrechung auf die richtige Verarbeitungseinrichtung gerichtet ist, um die Latenzzeit von dem Zeitpunkt, zu dem der Bedarf für ein Abschalten erfaßt wird, zu dem Zeitpunkt, wann der Datenfluß tatsächlich angehalten wird, zu minimieren. Folglich muß das Betriebssystem das Graphikhardwaregerät hinsichtlich dessen, zu welcher Verarbeitungseinrichtung gegenwärtig Daten geschickt werden, aktualisiert halten.

[0022] Wenn die Hardwareeingangspuffer leer genug sind, löst eine Neustartmarke eine weitere Unterbrechung aus, die zu der Verarbeitungseinrichtung geschickt wird. Wenn diese Unterbrechung von dem Betriebssystem empfangen wird, öffnet dasselbe erneut die Graphikhardware wirksam für einen Datenverkehr. Der Graphikprozeß mit höchster Priorität, der blockiert war, wird geweckt und ein Zugriff auf das Graphikhardwaregerät wird gewährt.

[0023] Durch Verwenden der Kombination eines Verlangsamens und Abschaltens des Datenflusses zu dem Graphikhardwaregerät, wird ein optimales Verhalten erreicht.

[0024] Ein Vorteil der Vorrichtung und des Verfahrens des virtuellen Gerätezugriffs gemäß der vorliegenden Erfindung besteht darin, daß dieselben die gerätabhängige Steuerung in einem Anwenderebenecode und nicht zusammen mit dem Betriebssystem plazieren. Dies erhöht die Flexibilität der Graphiksoftwarebibliotheken.

[0025] Ein Vorteil der Vorrichtung und des Verfahrens des virtuellen Gerätezugriffs gemäß der vorliegenden Erfindung besteht darin, daß dieselben Vorkehrungen für die Steuerung des Datenflusses ohne einen zugeordneten Zusatzaufwand in der Graphik-API treffen.

[0026] Ein Vorteil der Vorrichtung und des Verfahrens des virtuellen Gerätezugriffs gemäß der vorliegenden Erfindung besteht darin, daß dieselben eine Konkurrenz zwischen den Anwendungen beseitigen, die einen Zugriff auf das Graphikhardwaregerät versuchen.

[0027] Ein Vorteil der Vorrichtung und des Verfahrens des virtuellen Gerätezugriffs gemäß der vorliegenden Erfindung besteht darin, daß dieselben für einen Zugriff ohne Zusatzaufwand auf das Graphikelement sorgen.

[0028] Weitere Merkmale und Vorteile der vorliegenden Erfindung werden bei einer Prüfung der folgenden Zeichnungen und der detaillierten Beschreibung für einen Fachmann offensichtlich. Es ist beabsichtigt, daß alle zusätzlichen Merkmale und Vorteile, die hierin aufgenommen sind, sich in dem Bereich der vorliegenden Erfindung befinden, der durch die Ansprüche definiert ist.

Ausführungsbeispiel

[0029] Bevorzugte Ausführungsbeispiele der vorliegenden Erfindung werden nachfolgend unter Bezugnahme auf die begleitenden Zeichnungen näher erläutert. Es zeigen:

[0030] Fig. 1 ein Blockdiagramm eines Computersystems, das für eine Implementierung der vorliegenden Erfindung geeignet ist;

[0031] Fig. 2 ein Blockdiagramm der Architektur und der Funktionalität der Komponenten eine Vorrichtung eines virtuellen Gerätezugriffs, das für einen Betrieb auf dem Computersystem von Fig. 1 geeignet ist; und

[0032] Fig. 3A und Fig. 3B Ablaufdiagramme, die eine Verfahren des virtuellen Gerätezugriffs der vorliegenden Erfindung darstellen.

[0033] Die vorliegende Erfindung sorgt für ein System und ein Verfahren für einen virtuellen Gerätezugriff in einem Computersystem, das ermöglicht, daß die Graphikanwendungsprogrammschnittstellenbibliothek (API) ohne Leistungseinbußen pro Transaktion auf ein Graphikhardwaregerät zugreifen kann. Dies ist insbesondere aufgrund des Zusatzaufwandes, der einem Verriegeln und einem Entriegeln des Graphikhardwaregeräts mit jedem Bibliotheksaufruf zugeordnet ist, mit Scheitelpunkt-basierten APIs nützlich. Außerdem ist mit dem virtuellen Gerätezugriff der vorliegenden Erfindung keine gerätspezifische Kernunterstützung von dem Betriebssystem erforderlich. Stattdessen ist eine gerätspezifische Unterstützung auf der Anwenderebene durch eine Signalhandhabungseinrichtung vorgesehen, die jeder Anwendung zugeordnet ist. Dies ermöglicht eine größere Flexibilität, eine einfachere Entwicklung und Wartung, und eine einfachere Produktverbreitung. Ein weiterer wichtiger Aspekt des virtuellen Gerätezugriffs der vorliegenden

Erfindung ist die automatische Steuerung über den Datenfluß. Die vorliegende Erfindung beinhaltet einen neuen Lösungsansatz zum Steuern des Datenflusses von der Graphik-API zu der Graphikhardware, wodurch der Datenfluß im wesentlichen ohne zugeordneten Zusatzaufwand bei der Graphik-API automatisch verlangsamt oder angehalten wird.

[0034] Bezugnehmend nun auf die Zeichnungen ist in Fig. 1 ein Computersystem **10** dargestellt, das für eine Implementierung des Systems und Verfahrens eines virtuellen Gerätezugriffs der vorliegenden Erfindung geeignet ist, die es einer oder mehreren Anwendungen **12** ermöglicht, auf ein Graphikhardwaregerät **14** für eine Aufbereitung von Bildern auf einem Anzeigegerät **16** auf eine asynchrone Weise zuzugreifen.

[0035] Das Computersystem **10** weist eine oder mehrere herkömmliche Verarbeitungseinrichtungen **18** auf, von denen jede vorzugsweise einen Mehrprogrammbetrieb (Multitasking) ausführen kann, von denen jede über eine Systemschnittstelle **20** mit weiteren Geräten in dem Computersystem **10** kommuniziert. Zusätzlich ist ein Speicherbus **22** vorgesehen, um die Verarbeitungseinrichtungen) **18** und einen Speicher **24** zu verbinden, um die Übertragungsgeschwindigkeit zwischen dem Speicher und der/den Verarbeitungseinrichtung(en) **18** zu erhöhen. Ein Eingabegerät **26**, beispielsweise eine Tastatur oder eine Maus, wird verwendet, um Daten von einem Anwender des Computersystems **10** einzugeben. Das Graphikhardwaregerät **14** weist typischerweise eine Graphikbeschleunigerplatine und einen Graphikspeicher (Rahmenpuffer) zum Verarbeiten von Bilddaten auf, die auf dem Anzeigegerät **16** angezeigt werden. Der Graphikbeschleuniger des Graphikhardwaregeräts **14** ist vorzugsweise als Graphikpipeline zum Durchführen einer oder mehrerer der folgenden Aktionen konfiguriert: eine Transformation, eine Beleuchtung oder Schattenerzeugung, ein Abschneiden, eine perspektivische Unterteilung und eine Abtastungsumwandlung. Schließlich ist eine Festplatte **28** vorgesehen, um die Speicherkapazität des Computersystems **10** zu erhöhen.

[0036] In dem Speicher **24** können mittels eines herkömmlichen Betriebssystems **30** eine oder mehrere der Anwendungen **12** gleichzeitig ausgeführt werden. Um mit dem Graphikhardwaregerät **14** zu kommunizieren, ist jede Anwendung mit einer Graphik-API-Bibliothek **32** verbunden, derart, daß jede Anwendung **12** ihre eigene Graphik-API-Bibliothek **32** in ihrem virtuellen Adressenraum für eine Schnittstellenbildung mit dem Graphikhardwaregerät **14** aufweist. Die Graphik-API-Bibliothek **32** weist im wesentlichen Gerätetreiber auf, die von der Anwendung aufgerufen werden, um eine bestimmte Funktionalität auf dem Graphikhardwaregerät **14** durchzuführen. Insbesondere übersetzt die Graphik-API-Bibliothek **32** die API-Aufrufe in Graphikhardwarezugriffe.

[0037] Eine Graphikhardwarezugriffsvorrichtung **34** liefert die Einrichtung, mittels welcher die Graphik-API-Bibliothek **32** auf das Graphikhardwaregerät **14** zugreift. Die Graphikhardwarezugriffsvorrichtung **34** befindet sich in dem virtuellen Speicherbereich der Anwendungen **12** und ist mit dem Graphikhardwaregerät **14** direktspeicherkonform, um eine direkte Verbindbarkeit zwischen der Graphik-API-Bibliothek und dem Graphikhardwaregerät **14** zu liefern.

[0038] Bezugnehmend auf Fig. 2 sind die Architektur und die Funktionalität des Systems und Verfahrens für einen virtuellen Gerätezugriff der vorliegenden Erfindung dargestellt. Ein System **38** eines virtuellen Gerätezugriffs gemäß der vorliegenden Erfindung weist eine Hardwaresteuerungsverwaltungseinrichtung **40**, die in dem Betriebssystem **30** konfiguriert ist, eine Signalhandhabungseinrichtung **42**, die in dem virtuellen Speicherraum der Graphik-API-Bibliothek **32** jeder Anwendung **12** konfiguriert ist, und einen gemeinsam verwendeten Speicher **44** auf.

[0039] Die Hardwaresteuerungsverwaltungseinrichtung **40** ist als Gerätereiber in dem Betriebssystem **30** implementiert und umfaßt eine Steuerlogik, die verwaltet, welche Anwendung **12** einen Zugriff auf das Graphikhardwaregerät **14** hat, und wann ein Zugriff von einer Anwendung **12** zu einer anderen Anwendung **12** umgeschaltet werden soll. Bei dem bevorzugten Ausführungsbeispiel liefert die Hardwaresteuerungsverwaltungseinrichtung **40** eine gerätunabhängige Steuerung, die entworfen ist, um einen Zugriff auf eine beliebige Anwendung **12** zu geben, die versucht, auf das Graphikhardwaregerät **14** zuzugreifen, es sei denn, eine Anwendung, die gegenwärtig auf das Graphikhardwaregerät **14** zugreift, befindet sich in einem kritischen Abschnitt. Zum Zweck der vorliegenden Offenbarung ist ein kritischer Abschnitt ein Punkt bei der Datenverarbeitung, an dem das Graphikhardwaregerät **14** nicht unterbrochen werden sollte, wie es für einen Fachmann offensichtlich ist. Durch Entfernen des Bedarfs nach einer gerätspezifischen Unterstützung in dem Betriebssystem **30** für eine Implementierung des Systems **38** für einen virtuellen Gerätezugriff der vorliegenden Erfindung muß das Betriebssystem **30** nicht dynamisch mit dem gerätspezifischen Code geladen werden.

[0040] Die Hardwaresteuerungsverwaltungseinrichtung **40** steuert über die Graphikhardwarezugriffsvorrichtung (Fig. 1) jeder Anwendung **12** den Zugriff auf das Graphikhardwaregerät **14**. Die Graphikhardwarezugriffsvorrichtung **34** jeder Anwendung **12** ist in Fig. 2 als Kombination einer Verbindung **34'** und eines Umschalters **34"** dargestellt, die die jeweiligen Graphik-APIs **32** und das Graphikhardwaregerät **14** direkt verbinden. Die Umschalter **34"** werden von der

Hardwaresteuerungsverwaltungseinrichtung **40** über Befehlsverbindungen **48** gesteuert und sind vorgesehen, um die Verbindungen **34'** gemäß dem Verfahren der vorliegenden Erfindung, wie es im vorhergehenden beschrieben wurde, freizugeben oder zu sperren. Demgemäß haben die Graphik-APIs über die Graphikhardwarezugriffsvorrichtung **34** einen direkten Zugriff auf das Graphikhardwaregerät **14** zum Übertragen von Daten, wie z. B. Farbe, Textur, Perspektive usw., und zum Übertragen von Graphikhardwarezustandsinformationen.

[0041] Die Graphikhardwarezugriffsvorrichtung **34** (d. h. die Elemente **34'** und **34"** in Fig. 2) jeder Anwendung ist durch Schutzidentifizierer (PID; PID = Protection Identifier = Schutzidentifizierer) geschützt, die in Hinblick auf virtuelle Speicheradressen in der Industrie bekannt sind. Die PIDs, die den Graphikhardwarezugriffsvorrichtungen **34** zugeordnet sind, sind konfiguriert, um jedes mal einen Schutzfehler zu erzeugen, wenn eine Anwendung **12** versucht, auf das Graphikhardwaregerät **14** zuzugreifen, wenn auf dasselbe gegenwärtig kein Zugriff möglich ist. Im wesentlichen kann der Umschalter **34"** als die Schutzidentifiziererhardware gedacht sein. Der Umschalter **34"** läßt entweder den Zugriff durch oder erzeugt an dem Hardwaresteuerblock **40** einen Fehler. Der Fehler wird über die Verbindung **48** übermittelt, während Verbindungen **46** für ein Einrichten und/oder weitere Verwaltungsübermittlungen zwischen der Graphik-API-Bibliothek **32** und der Hardwaresteuerungsverwaltungseinrichtung **40** verwendet werden.

[0042] Die Signalhandhabungseinrichtung **42** jeder Anwendung **12** stellt die hardwaregeräteabhängige Steuerung, die dem Hardwarezugriff zugeordnet ist, bereit. Die Signalhandhabungseinrichtung **42** führt die Funktionalität des Sicherns des Graphikhardwarezustands der Anwendung **12**, die gegenwärtig auf das Graphikhardwaregerät **14** zugreift, und des Wiederherstellens des Graphikhardwarezustands der Anwendung **12** durch, die versucht, auf das Graphikhardwaregerät **14** zuzugreifen. Dieser Prozeß wird als Graphikkontextumschaltung bezeichnet. Als wichtiges Merkmal der vorliegenden Erfindung wird die Graphikkontextumschaltung durch die Signalhandhabungseinrichtung auf eine Art und Weise durchgeführt, die für jede Anwendung **12** transparent ist, damit kein Zusatzaufwand in der Graphik-API-Bibliothek **32** verursacht wird, und derart, daß eine Zugriffskonkurrenz für jede Anwendung **12** transparent ist.

[0043] Die Signalhandhabungseinrichtung **42** kommuniziert über eine Verbindung **52** mit der Hardwaresteuerungsverwaltungseinrichtung **40**. Die Verbindung **52** wird verwendet, um Befehlssignale zwischen der Signalhandhabungseinrichtung **42** und der Hardwaresteuerungsverwaltungseinrichtung **40** für eine solche Aufgabe, wie das Einleiten einer Graphikkontextumschaltung, zu übertragen oder um den Abschluß einer Graphikkontextumschaltung mitzuteilen. Zusätzlich überprüft die Signalhandhabungseinrichtung **42** den Fenstergeometriezustand auf Änderungen und aktualisiert entsprechend den Hardwaregraphikzustand.

[0044] Der gemeinsam verwendete Speicher **44** liefert eine gemeinsame Speicherposition, auf die jede Anwendung **12** zugreifen kann. Auf den gemeinsam verwendeten Speicher **44** kann über jeweilige Verbindungen **54** von jeder Anwendung zum Sichern und Wiedergewinnen der Graphikhardwarezustände der Anwendungen für eine Verwendung bei einer Graphikkontextumschaltung zugegriffen werden. Zusätzlich umfaßt der gemeinsam verwendete Speicher ein globales, gemeinsam verwendetes Objekt, das als der Graphikkontextzeiger (nicht gezeigt) bezeichnet wird, der anzeigt, welche Anwendung gegenwärtig auf das Graphikhardwaregerät **14** zugreift. Diese Informationen in dem Graphikkontextzeiger werden von der Signalhandhabungseinrichtung **42** verwendet, die eine Graphikkontextumschaltung durchführt, derart, daß die Signalhandhabungseinrichtung den Graphikhardwarezustand der Anwendung, die gegenwärtig auf das Graphikhardwaregerät **14** zugreift, lesen kann und daraufhin diese Zustandsinformationen in einer Speicherposition innerhalb des gemeinsam verwendeten Speichers **44** speichern kann, die der gegenwärtigen Anwendung **12** zugeordnet ist. Folglich wird diese Anwendung wissen, wo sich deren Zustandsinformationen in dem gemeinsam verwendeten Speicher **44** befinden, wenn dieselbe erneut einen Zugriff auf das Graphikhardwaregerät **14** erlangt. Es sollte bekannt sein, daß die Graphikhardwarezustände über die Graphikhardwarezugriffsvorrichtung **34**, die der Anwendung **12** zugeordnet ist, die versucht, auf die Graphikhardware zuzugreifen, sowohl in das Graphikhardwaregerät **14** geschrieben sind als auch von demselben gelesen werden.

[0045] Zusätzlich kann die Hardwaresteuerungsverwaltungseinrichtung **40** über eine gerätabhängige Schnittstelle **56** einfache Operationen bezüglich des Graphikhardwaregerätes **14** durchführen. Die gerätabhängige Schnittstelle **56** ist als Teil des Graphikhardwaregeräts **14** konfiguriert und über eine Verbindung 58 mit der Hardwaresteuerungsverwaltungseinrichtung **40** verbunden. Die gerätabhängige Schnittstelle **56** weist einen gerätabhängigen Nur-Lese-Speicher (ROM; ROM = Read Only Memory = Nur-Lese-Speicher) mit einer Kern-zu-Hardware-Schnittstelle auf. Demgemäß wird ein in dem ROM bereitgestellter Code verwendet, um die Steuersignale von der Hardwaresteuerungsverwaltungseinrichtung **40** zu interpretieren, derart, daß dieselben von dem Graphikhardwaregerät **14** ausgeführt werden können. Beispiele dieser Operationen, die über die gerätabhängige Schnittstelle **56** durchgeführt werden, sind einfache Terminaloperationen (d.

h. Zeichnen eines Zeichens oder Scrollen eines Bildschirms) zusätzlich zu Unterbrechungssteuerungs- und Gerätestatusfunktionen.

[0046] Bezugnehmend auf Fig. 2 ist eine Flußsteuervorrichtung **62** als Teil des Graphikhardwaregeräts **14** zum automatischen Verlangsamen oder Anhalten des Datenfluß zu dem Graphikhardwaregerät **14** vorgesehen, ohne daß ein Zusatzaufwand in der Graphik-API verursacht wird. Da die Zeit, die für das Graphikhardwaregerät **14** erforderlich ist, um die Bilddaten für eine bestimmte Anwendung **12** zu verarbeiten, abhängig von den Daten und der Verarbeitung, die mit diesen Daten durchgeführt wird, um mehrere Größenordnungen variieren kann, ist es möglich, daß die Eingangsdatenpuffer (nicht gezeigt) des Graphikhardwaregeräts **14** zu voll werden können, um weitere Eingangsdaten zu akzeptieren. Wie es in dem Hintergrund-Abschnitt erörtert wurde, haben frühere Lösungen dieses Problems einen wesentlichen Zusatzaufwand erfordert und/oder in dem Verlust von Daten resultiert. Die Steuerflußvorrichtung **62** der vorliegenden Erfindung überwacht in den Eingangspuffern des Graphikhardwaregeräts **14** jedoch eine niedrige Füllstandsmarke und überwacht, wann die Eingangspuffer die niedrige Füllstandsmarke erreichen, wobei diese Steuerflußvorrichtung **62** auslöst, daß die Schnittstellensteuerlogik des Graphikgeräts **14** damit beginnt, auf die Transaktionen über die Graphikhardwarezugriffsvorrichtung **34** langsamer zu antworten. Diese Verzögerungen ergeben eventuell, daß die Verarbeitungseinrichtung **18** die Daten mit einer niedrigeren Rate schreibt, wodurch die Datenübertragungsrate wirksam gedrosselt wird, ohne daß ein zusätzlicher Zusatzaufwand in der Graphik-API-Bibliothek **32** verursacht wird.

[0047] Falls sich jedoch der Eingangsdatenpuffer weiter zu dem Punkt füllt, bei dem derselbe eine hohe Füllstandsmarke erreicht, schickt die Flußsteuervorrichtung **62** ein zweites Steuersignal direkt an die Verarbeitungseinrichtung **18**, die die Daten schreibt, um den Datenfluß zu unterbrechen. Das Betriebssystem **30** erfaßt diese Unterbrechung und erkennt dieselbe als Anzeige, daß die Graphikhardwareeingangspuffer einen Rückstau aufweisen, und wird folglich für eine Anwendung die Erlaubnis zum Zugreifen auf das Graphikhardwaregerät **14** entfernen. Es sollte beachtet werden, daß die Flußsteuervorrichtung **62** verfolgen muß, welche Verarbeitungseinrichtung **18** gegenwärtig Daten in das Graphikhardwaregerät **14** schreibt, damit das zweite Signal zu der korrekten Verarbeitungseinrichtung **18** in einer Umgebung mit mehreren Verarbeitungseinrichtungen geschickt wird.

[0048] Die vorliegende Erfindung sorgt ferner für einen virtuellen Gerätezugriff und kann als Verfahren für einen virtuellen Gerätezugriff konzipiert werden, wodurch eine Anwendung **12**, die auf einem Computersystem **10** arbeitet, ohne Leistungseinbußen pro Transaktion auf das Graphikhardwaregerät **14** zugreifen kann. Wie es im vorhergehenden erörtert wurde, besteht ein wichtiges Merkmal dieses Verfahrens darin, daß dasselbe keine gerätspezifische Unterstützung von dem Betriebssystem **30** benötigt. Die gerätspezifische Unterstützung wird von der Signalhandhabungseinrichtung **42** vorgesehen, die der API-Bibliothek **12** zugeordnet ist. Es wird zum Zweck der Offenbarung des bevorzugten Ausführungsbeispiels angenommen, daß die Steuerlogik, die von der Hardwaresteuerungsverwaltungseinrichtung **40** bereitgestellt wird, derart beschaffen ist, daß die Hardwaresteuerungsverwaltungseinrichtung **40** einen Hardwarezugriff auf eine beliebige Anwendung, die einen Zugriff versucht, für zumindest eine minimale Graphikzeitscheibe gewähren wird, es sei denn, die Anwendung, die gegenwärtig auf das Graphikhardwaregerät zugreift, befindet sich in einem kritischen Abschnitt. Wie es für einen Fachmann offensichtlich ist, gibt es Zeiten, bei denen ein Zugriff auf das Graphikhardwaregerät **14** nicht unterbrochen werden sollte, und daß diese Zeiten als kritische Abschnitte besonders geschützt sind, wie es im vorhergehenden beschrieben wurde.

[0049] Bezugnehmend nun auf die Fig. 3A und Fig. 3B ist ein Beispiel eines virtuellen Gerätezugriffs gemäß dem Verfahren der vorliegenden Erfindung dargestellt, wodurch eine Anwendung **12** versucht, auf das Graphikhardwaregerät **14** zuzugreifen. Zu Anfang, d. h. in einem Block **60**, versucht eine Anwendung **12** über die Graphikhardwarezugriffsvorrichtung **34**, die dieser Anwendung zugeordnet ist, auf das Graphikhardwaregerät **14** zuzugreifen. Wie es im vorhergehenden erörtert wurde, wird ein Schutzidentifiziererfehler (PID-Fehler), der jeder Graphikhardwarezugriffsvorrichtung **34** zugeordnet ist, erzeugt, wenn eine Anwendung versucht, auf das Graphikhardwaregerät **14** zuzugreifen. Wenn die Anwendung **12** versucht, auf das Graphikhardwaregerät **14** zuzugreifen, wird demgemäß ein PID-Fehler erzeugt, wobei die Anwendung blockiert wird, d. h. der Anwendungsprozeß kann die Ausführung nicht fortsetzen, bis der Fehler gelöst ist.

[0050] Die Hardwaresteuerungsverwaltungseinrichtung **40** erkennt den PID-Fehler als eine Anwendung, die versucht, auf das Graphikhardwaregerät **14** zuzugreifen. Als Reaktion führt die Hardwaresteuerungsverwaltungseinrichtung **40** eine Bestimmung durch, ob sich die Anwendung, die gegenwärtig auf das Graphikhardwaregerät **14** zugreift, in einem kritischen Abschnitt befindet, wie es durch einen Entscheidungsblock **62** angezeigt ist. Falls sich die Anwendung, die gegenwärtig auf das Graphikhardwaregerät **14** zugreift, in einem kritischen Abschnitt befindet, blockiert die Hardwaresteuerungsverwaltungseinrichtung **40** den Anwendungsprozeß, der darauf wartet, daß der kritische Abschnitt freigegeben wird, wie

es in einem Block **64** angezeigt ist. Als nächstes, d. h. in einem Block **66**, wählt die Hardwaresteuerungsverwaltungseinrichtung **40** eine wartende Anwendung aus, um einen Zugriff auf das Graphikhardwaregerät **14** zu empfangen, sobald der kritische Abschnitt freigegeben worden ist. Ausgehend von einem Block **66** beginnt der Prozeß, der im vorhergehenden beschrieben wurde, wieder bei dem Block **62**, wobei die Hardwaresteuerungsverwaltungseinrichtung **40** überprüft, um zu erkennen, ob sich die gegenwärtige Anwendung in einem kritischen Abschnitt befindet.

[0051] Falls jedoch die Hardwaresteuerungsverwaltungseinrichtung **40** in diesem Entscheidungsblock **62** bestimmt, daß sich die Anwendung **12**, die gegenwärtig auf das Graphikhardwaregerät **14** zugreift, nicht in einem kritischen Abschnitt befindet, dann gibt die Hardwaresteuerungsverwaltungseinrichtung **40** die Graphikhardwarezugriffsvorrichtung **34** der Anwendung frei, die im vorhergehenden auf das Graphikhardwaregerät **14** zugegriffen hat, wie es durch einen Block **68** von Fig. 3B angezeigt ist. Als nächstes schickt die Hardwaresteuerungsverwaltungseinrichtung **40** ein Steuersignal zu der Signalhandhabungseinrichtung **42** der Anwendung **12**, die einen Zugriff versucht, wodurch diese Signalhandhabungseinrichtung **42** angewiesen wird, sowohl eine Graphikkontextumschaltung als auch eine Aktualisierung der Fenstergeometriezustände durchzuführen, wie es durch einen Block **70** angezeigt ist.

[0052] In einem Block **72** führt die Signalhandhabungseinrichtung **42** der Anwendung **12**, die einen Zugriff versucht, eine Graphikkontextumschaltung durch. Dies beinhaltet, daß die Signalhandhabungseinrichtung **42** den Hardwarezustand des Graphikhardwaregeräts **14** für die Anwendung sichert, die gegenwärtig auf das Graphikhardwaregerät **14** zugreift, derart, daß die Anwendung wieder mit der Verarbeitung der Graphikdaten an dem Ort beginnen kann, an dem dieselbe angehalten würde, derart, daß dieselbe einen Zugriff an die Anwendung **12**, die versucht, einen Zugriff zu erlangen, abtreten könnte. Wie es im vorhergehenden erwähnt wurde, kann eine Graphikkontextumschaltung lediglich dann auftreten, wenn die Anwendung, die gegenwärtig auf das Graphikhardwaregerät **14** zugreift, sich nicht in einem kritischen Abschnitt befindet, wie es in dem Block **66** bestimmt wird. Der Schritt des Sicherns des Hardwarezustandes umfaßt das Auslesen des Hardwarezustandes von dem Graphikhardwaregerät **14** über die Graphikhardwarezugriffsvorrichtung **34** der anfordernden Anwendung **12** und das Speichern dieser Informationen in dem gemeinsam verwendeten Speicher **44**. Die Signalhandhabungseinrichtung **42** muß diese Informationen in einer bestimmten Position in einem gemeinsam verwendeten Speicher **44** plazieren, der der Anwendung zugeordnet ist, die im vorhergehenden auf das Graphikhardwaregerät **14** zugegriffen hat, derart, daß die Anwendung weiß, wo sich ihre Hardwarezustandsinformationen befinden, wenn dieselbe versucht, erneut einen Zugriff auf das Graphikhardwaregerät **14** zu erlangen. Diese Position wird von dem Graphikkontextzeiger in dem gemeinsam verwendeten Speicher **44** geliefert, welcher anzeigt, welcher Anwendungsgraphikkontext gegenwärtig in das Graphikhardwaregerät **14** geladen wird, und folglich anzeigt, wo die Hardwarezustandsinformationen zu sichern sind, wenn eine weitere Anwendung, die versucht, auf das Graphikhardwaregerät **14** zuzugreifen, eine Graphikkontextumschaltung durchführt. Bei dem bevorzugten Ausführungsbeispiel wird das Auslesen des Hardwarezustandes von dem Graphikhardwaregerät **14** vorzugsweise durch einen direkten Speicherzugriff (DMA; DMA = Direct Memory Access = direkter Speicherzugriff) ausgeführt, obwohl es offensichtlich ist, daß es zahlreiche weitere geeignete Verfahren gibt, beispielsweise über ein direktes Hardwareauslesen.

[0053] Der zweite Teil der Graphikkontextumschaltung stellt den Hardwarezustand der Anwendung **12**, die versucht, auf das Graphikhardwaregerät **14** zuzugreifen, wieder her. Der Hardwarezustand der Anwendung **12**, die einen Zugriff versucht, wird von der Signalhandhabungseinrichtung **42** aus dem gemeinsam verwendeten Speicher **44** wiedergewonnen und über die Graphikhardwarezugriffsvorrichtung **34** in das Graphikhardwaregerät **14** geschrieben. Vorzugsweise wird wie bei dem Sicherungsschritt der Graphikkontextumschaltung der Wiederherstellschritt mit einem DMA durchgeführt, obwohl es offensichtlich ist, daß andere Verfahren genauso geeignet sein können. Zusätzlich aktualisiert die Signalhandhabungseinrichtung **42** falls nötig den Fenstergeometriezustand in der Hardware. In einem Block **72** schickt die Signalhandhabungseinrichtung **42** der Anwendung, die einen Zugriff versucht, ein Rücksprungsteuersignal zu der Hardwaresteuerungsverwaltungseinrichtung **40**, daß die Graphikkontextumschaltung durchgeführt und die Fenstersystemwerte gegebenenfalls aktualisiert worden sind.

[0054] Als Reaktion auf das Rücksprungsteuersignal gibt die Hardwaresteuerungsverwaltungseinrichtung **40** die Erlaubnis an die Anwendung **12**, die einen Zugriff versucht, auf das Graphikhardwaregerät **14** zuzugreifen, und gibt die Graphikhardwarezugriffsvorrichtung **34** der anfordernden Anwendung frei, wie es in einem Block **74** angezeigt ist. Wie es in Fig. 2 dargestellt ist, wird dies durch ein Schließen des Umschalters **34"** über die Verbindung **48** dargestellt.

[0055] Um die Wirksamkeit, die von dem virtuellen Gerätezugriff erreicht wird, zu erhöhen, liefert die vorliegende Erfindung ferner eine automatische Datenflußsteuerung von der Graphik-API **32** zu dem Graphikhardwaregerät **14**. Insbesondere verhindert das

Flußsteuerschema der vorliegenden Erfindung, daß die Anwendung den Status der Eingangsdatenpuffer des Graphikhardwaregeräts **14** auf einer kontinuierlichen Basis überprüfen muß, um zu bestimmen, ob die Datenpuffer mehr Daten empfangen können, was als aufwendige, jedoch notwendige Aufgabe bei den bekannten Schemata beurteilt werden kann.

[0056] Gemäß dem Flußsteuerschema der vorliegenden Erfindung, wie es durch die Flußsteuervorrichtung **62** implementiert ist, wird die Anwendung automatisch verzögert, wenn das Graphikhardwaregerät **14** die Daten nicht schnell genug verarbeiten kann, um mit der Anwendung Schritt zu halten, ohne daß ein Datenverlust riskiert wird. Im allgemeinen überwacht eine Flußsteuervorrichtung **62** eine Anzeige einer niedrigen Füllstandsmarke, die von dem Graphikhardwaregerät **14** geliefert wird, um zu erfassen, wann die Daten in den Eingangsdatenpuffern einen ersten vorbestimmten Pegel erreichen. Außerdem wird ferner eine hohe Füllstandsmarke von der Flußsteuervorrichtung **62** überwacht, um zu erfassen, wann die Daten in dem Dateneingangspuffer einen zweiten vorbestimmten Pegel erreichen.

[0057] Im wesentlichen beginnt die Flußsteuervorrichtung **62** automatisch damit, auf die Bustransaktionen langsamer zu antworten, wenn die Eingangsdatenpuffer des Graphikhardwaregerätes **14** bis zu der niedrigen Füllstandsmarke gefüllt sind. Die Wirkung dieses ersten Steuersignals breitet sich eventuell zurück zu der Verarbeitungseinrichtung **18** aus, die die Daten in das Graphikhardwaregerät **14** schreibt, wodurch die Datenrate wesentlich gedrosselt wird, ohne einen Zusatzaufwand in der Bibliothek der Graphik-API **32** hinzuzufügen oder externe Unterbrechungen an der Verarbeitungseinrichtung zu erzeugen.

[0058] Falls die Datenrate für das Graphikhardwaregerät **14** noch zu schnell ist und die Daten in dem Eingangsdatenpuffer die hohe Füllstandsmarke erreichen, wird eine gerichtete Unterbrechung oder ein zweites Steuersignal zu der Verarbeitungseinrichtung **18** geschickt, die die Daten schreibt, um die Verarbeitungseinrichtung **18** von dieser Tatsache zu unterrichten. Als Ergebnis wird verhindert, daß die Verarbeitungseinrichtung **18** weitere Daten zu dem Graphikhardwaregerät **14** schickt. Die Daten in dem Eingangsdatenpuffer werden eventuell unter die niedrige Füllstandsmarke fallen, wobei die Flußsteuervorrichtung ein Neustartsignal zu der Verarbeitungseinrichtung **18** schicken wird, das anzeigt, daß die Anwendung **12** das Schicken der Daten zu dem Graphikhardwaregerät **14** wieder aufnehmen kann.

[0059] Um dieses Merkmal der Erfindung in einer Umgebung mit mehreren Verarbeitungseinrichtungen zu implementieren, muß das Graphikhardwaregerät **14** wissen, welche Verarbeitungseinrichtung **18** ihm gegenwärtig Daten schickt. Ohne diese Informationen muß die Unterbrechung der hohen Füllstandsmarke alle Verarbeitungseinrichtungen **18** unterbrechen, um sicherzustellen, daß die Daten rechtzeitig angehalten werden. Falls die Verarbeitungseinrichtung **18**, die die Daten sendet, nicht angehalten wird, bevor die Kapazität der Eingangsdatenpuffer überschritten ist, werden Daten verloren, was nicht akzeptabel ist. Wie es für einen Fachmann offensichtlich sein würde, kann dies über einen Verarbeitungsplaner in dem Betriebssystem erreicht werden, der dafür verantwortlich ist, die Flußsteuervorrichtung **62** mit der Identifizierung der korrekten Verarbeitungseinrichtung **18** zu aktualisieren, sowie die Graphikanwendungen ausgeführt werden.

Patentansprüche

1. Vorrichtung für einen virtuellen Gerätezugriff zum Vorsehen einer Verriegelung ohne Zusatzaufwand eines Graphikhardwaregeräts (**14**) eines Computersystems (**10**) für Anwendungen (**12**), die auf dem Computersystem (**10**) laufen, mit folgenden Merkmalen:
einer Hardwaresteuerungsverwaltungseinrichtung (**40**) zum Erfassen, wann eine der Anwendungen (**12**) versucht, auf das Graphikhardwaregerät (**14**) zuzugreifen, und zum Freigeben und Sperren des Zugriffs auf das Graphikhardwaregerät (**14**);
einer Anwenderebene-Signalhandhabungseinrichtung (**42**), die jeder der Anwendungen (**12**) zugeordnet ist, wobei jede der Anwenderebene-Signalhandhabungseinrichtungen (**42**) zum Durchführen einer Graphikkontextumschaltung konfiguriert ist, wobei, wenn eine der Anwendungen (**12**) versucht, auf das Graphikhardwaregerät (**14**) zuzugreifen, die Hardwaresteuerungsverwaltungseinrichtung (**40**) ein Signal an die Anwenderebene-Signalhandhabungseinrichtung (**42**) ausgibt, die der Anwendung zugeordnet ist, die versucht, auf das Graphikhardwaregerät zuzugreifen, wobei die Anwenderebene-Signalhandhabungseinrichtung, die das Signal von der Hardwaresteuerungsverwaltungseinrichtung empfangen hat, eine Graphikkontextumschaltung als Reaktion auf den Empfang des Signals von der Hardwaresteuerungsverwaltungseinrichtung durchführt; und
einem Shared-Memory-Betriebsmittel (**44**), auf das durch jede der Anwendungen (**12**) zugegriffen werden kann und das zum Speichern einer Identität der Anwendung (**12**) konfiguriert ist, die gegenwärtig auf das Graphikhardwaregerät (**14**) zugreift, für eine Verwendung durch jede der Anwenderebene-Signalhandhabungseinrichtungen (**42**), wenn die Graphikkontextumschaltung durchgeführt wird.

2. Vorrichtung gemäß Anspruch 1, das eine Graphikhardwarezugriffsvorrichtung (**34**) aufweist, die jeder der Anwendungen (**12**) zugewiesen ist und direkt in das Graphikhardwaregerät (**14**) abgebildet ist, zum

Handhaben direkter Kommunikationen jeweils zwischen den Anwendungen (**12**) und dem Graphikhardwaregerät (**14**).

3. Vorrichtung gemäß Anspruch 1, bei dem eine der Anwendungen (**12**) eine Graphikanwendungsschnittstellenbibliothek (**32**) zum Kommunizieren mit dem Graphikhardwaregerät (**14**) aufweist.

4. Vorrichtung gemäß Anspruch 1, bei dem die Hardwaresteuerungsverwaltungseinrichtung (**40**) eine Steuerungslogik zum Verwalten einer Konkurrenz und eines Zugriffs auf das Graphikhardwaregerät (**14**) durch die Anwendungen (**12**) liefert.

5. Vorrichtung gemäß Anspruch 1, bei dem die Graphikkontextumschaltung, die von den Signalhandhabungseinrichtungen (**42**) durchgeführt wird, ein Sichern eines ersten Graphikhardwarezustands einer ersten Anwendung (**12**), die gegenwärtig auf das Graphikhardwaregerät (**14**) zugreift, in dem Shared-Memory-Betriebsmittel (**44**) und ein Wiederherstellen zu dem Graphikhardwaregerät (**14**) eines zweiten Graphikhardwarezustandes einer zweiten Anwendung (**12**) umfaßt, die versucht, auf das Graphikhardwaregerät (**14**) zuzugreifen.

6. Vorrichtung gemäß Anspruch 1, das ferner eine Flußsteuerungsvorrichtung (**62**) zum Verwalten des Datenflusses von den Anwendungen (**12**) zu dem Graphikhardwaregerät (**14**) aufweist.

7. Vorrichtung gemäß Anspruch 2, bei dem die Hardwaresteuerungsverwaltungseinrichtung (**40**) die Graphikhardwarezugriffsvorrichtung (**34**), die jeder der Anwendungen (**12**) zugeordnet ist, freigibt und sperrt.

8. Vorrichtung gemäß Anspruch 2, bei dem jede der Graphikhardwarezugriffsvorrichtungen (**34**) einen Schutzidentifizierer aufweist, um der Hardwaresteuerungsverwaltungseinrichtung (**40**) zu signalisieren, wann eine jeweilige der Anwendungen (**12**) versucht, auf das Graphikhardwaregerät (**14**) zuzugreifen, und um zu identifizieren, welche der Anwendungen (**12**) versucht, auf das Graphikhardwaregerät (**14**) zuzugreifen.

9. Vorrichtung gemäß Anspruch 6, bei dem die Flußsteuervorrichtung (**62**) eine Erfassungseinrichtung zum Reduzieren einer Datenflußrate zu dem Graphikhardwaregerät (**14**) aufweist, wenn ein Eingangsdatenpuffer des Graphikhardwaregeräts (**14**) über einem ersten vorbestimmten Pegel gefüllt ist, und zum Senden eines zweiten Steuersignals zu der Hardwaresteuerungsverwaltungseinrichtung (**40**), um den Datenfluß zu dem Graphikhardwaregerät (**14**) anzuhalten, wenn der Eingangsdatenpuffer über einem zweiten vorbestimmten Pegel gefüllt ist.

10. Vorrichtung gemäß Anspruch 7, bei dem die Steuerlogik der Hardwaresteuerungsverwaltungseinrichtung (**40**) jeder Anwendung (**12**), die auf das Graphikhardwaregerät (**14**) zugreift, eine minimale Menge einer Verarbeitungszeit zugesteht.

11. Verfahren für einen virtuellen Gerätezugriff, das eine Verriegelung ohne Zusatzaufwand eines Graphikhardwaregeräts (**14**) eines Computersystems (**10**) für Anwendungen (**12**) liefert, die auf dem Computersystem (**10**) laufen, wobei das Verfahren folgende Schritte aufweist:
Erfassen, wann eine erste Anwendung (**12**) versucht, auf das Graphikhardwaregerät zuzugreifen;
Bestimmen, ob eine zweite Anwendung (**12**) gegenwärtig auf das Graphikhardwaregerät (**14**) in einem kritischen Abschnitt zugreift;
falls sich die zweite Anwendung (**12**) in dem kritischen Zustand befindet, blockiert eine Hardwaresteuerungsverwaltungseinrichtung (**40**) die erste Anwendung (**12**), bis der kritische Abschnitt freigegeben ist; und
falls sich die zweite Anwendung (**12**) nicht in dem kritischen Abschnitt befindet, dann werden folgende Schritte durchgeführt:
Durchführen einer Graphikkontextumschaltung zwischen der ersten und der zweiten Anwendung (**12**), wobei die Graphikkontextumschaltung durch eine Anwenderebene-Signalhandhabungseinrichtung (**42**) durchgeführt wird, und
Geben einer Erlaubnis an die erste Anwendung (**12**), um auf das Graphikhardwaregerät (**14**) zuzugreifen.

12. Verfahren gemäß Anspruch 11, das ferner den Schritt des Signalisierens an eine Signalhandhabungseinrichtung (**42**) aufweist, die der ersten Anwendung (**12**) zugeordnet ist, eine Graphikkontextumschaltung durchzuführen, falls sich die zweite Anwendung (**12**) nicht in dem kritischen Abschnitt befindet.

13. Verfahren gemäß Anspruch 11, bei dem der Schritt des Durchführens einer Kontextumschaltung folgende Schritte aufweist:
Sichern eines ersten Graphikhardwarezustands der zweiten Anwendung (**12**) in einem gemeinsam verwendeten Speicher (**44**); und
Wiederherstellen eines zweiten Graphikhardwarezustands der ersten Anwendung (**12**) in dem Graphikhardwaregerät (**14**).

14. Verfahren gemäß Anspruch 11, das ferner den Schritt des Sperrens eines Zugriffs der zweiten Anwendung (**12**) auf das Graphikhardwaregerät (**14**) aufweist, falls sich die zweite Anwendung (**12**) nicht in dem kritischen Abschnitt befindet.

15. Verfahren gemäß Anspruch 11, bei dem die Anwendung (**12**) einen direkten Zugriff auf das Graphikhardwaregerät (**14**) über eine Graphikhardware-

zugriffsvorrichtung (**34**) hat.

16. Verfahren gemäß Anspruch 11, bei dem der Schritt des Durchführens der Graphikkontextumschaltung ein Aktualisieren der Fenstersystemwerte umfaßt.

17. Verfahren gemäß Anspruch 11, das ferner den Schritt des Steuerns eines Datenflusses zu dem Graphikhardwaregerät (**14**) aufweist.

Es folgen 4 Blatt Zeichnungen

Anhängende Zeichnungen

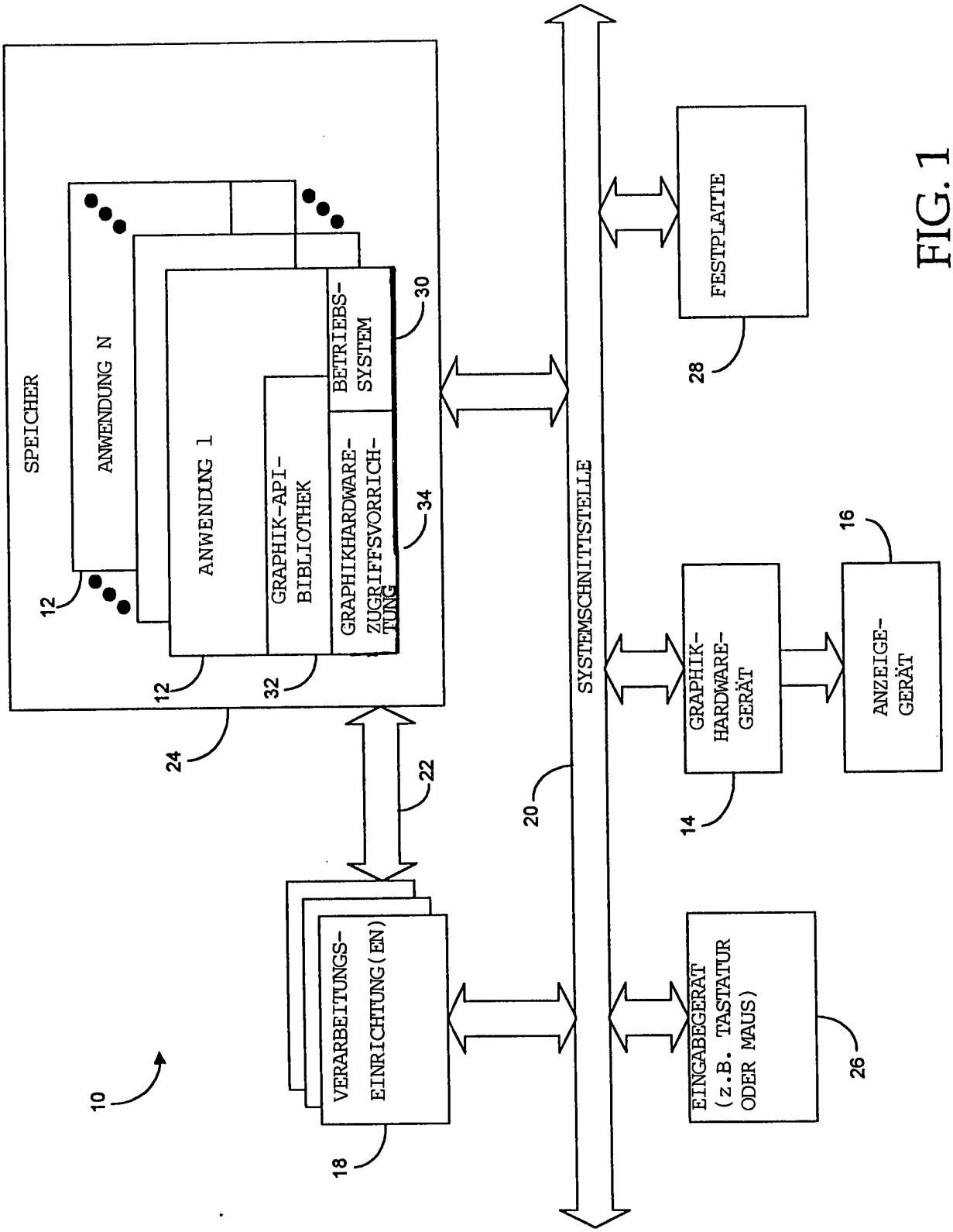


FIG. 1

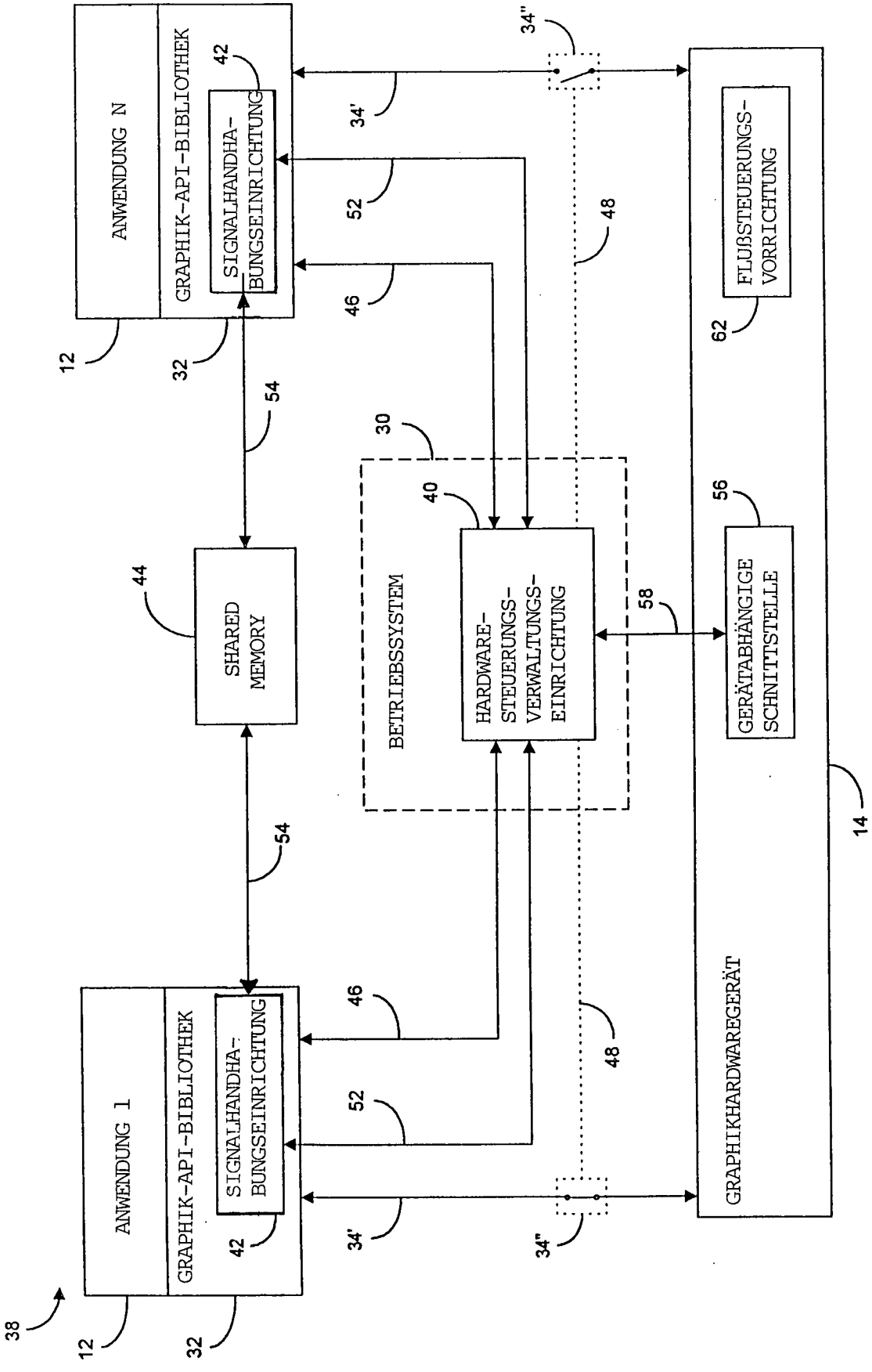

ANWENDUNG N
GRAPHIK-API-BIBLIOTHEK
SIGNALHANDHA-BUNGSEINRICHTUNG
42
34'
34"
52
48
46
FLUßSTEUERUNGS-VORRICHTUNG
62
12
32
54
30
40
56
44
SHARED MEMORY
BETRIEBSSYSTEM
HARDWARE-STEUERUNGS-VERWALTUNGS-EINRICHTUNG
58
GERÄTABHÄNGIGE SCHNITTSTELLE
14
ANWENDUNG 1
GRAPHIK-API-BIBLIOTHEK
SIGNALHANDHA-BUNGSEINRICHTUNG
GRAPHIKHARDWAREGERÄT
38

FIG 2

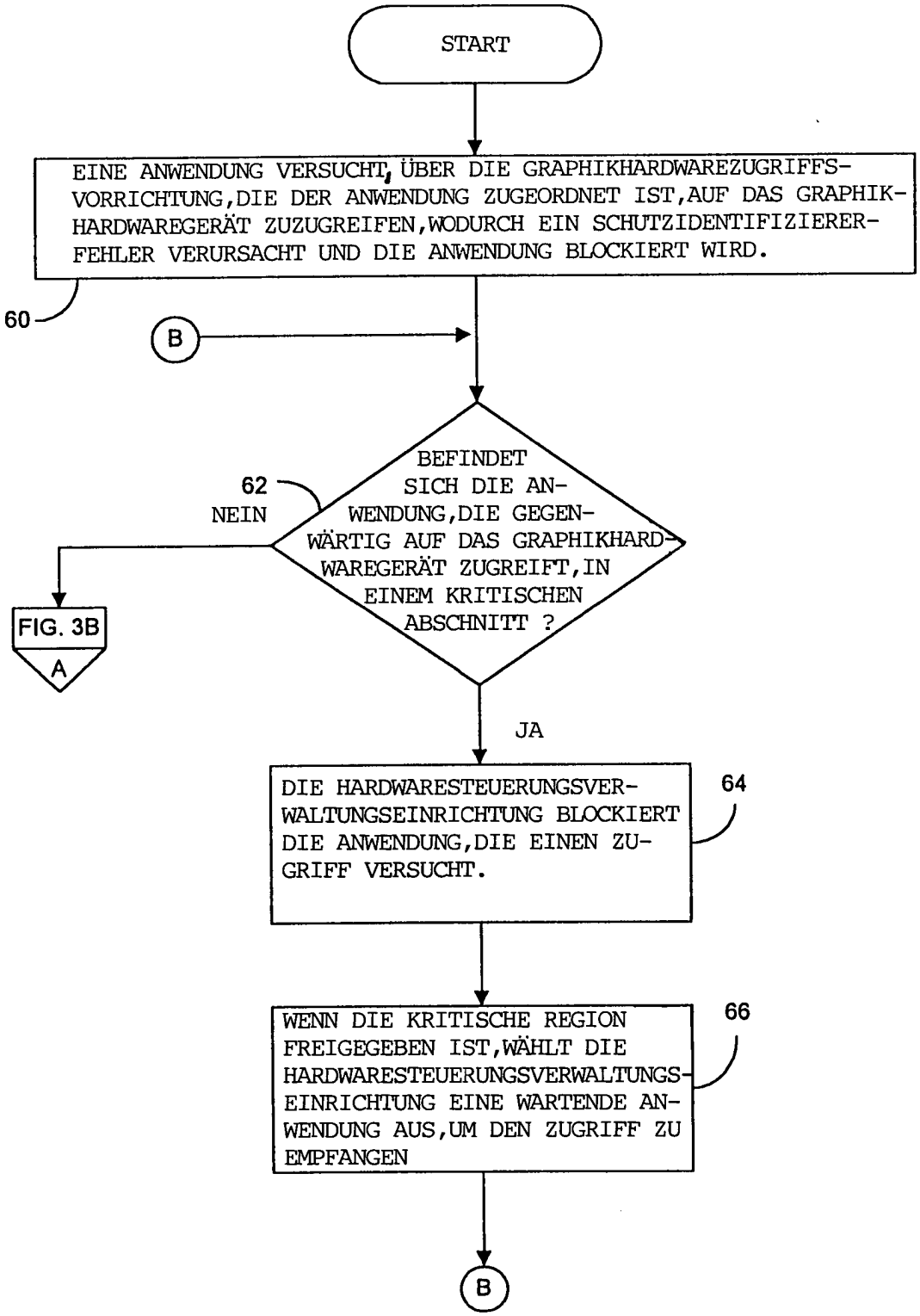

START
EINE ANWENDUNG VERSUCHT, ÜBER DIE GRAPHIKHARDWAREZUGRIFFS-VORRICHTUNG,DIE DER ANWENDUNG ZUGEORDNET IST,AUF DAS GRAPHIK-HARDWAREGERÄT ZUZUGREIFEN,WODURCH EIN SCHUTZIDENTIFIZIERER-FEHLER VERURSACHT UND DIE ANWENDUNG BLOCKIERT WIRD.
60
B
62
BEFINDET SICH DIE AN-WENDUNG,DIE GEGEN-WÄRTIG AUF DAS GRAPHIKHARD-WAREGERÄT ZUGREIFT,IN EINEM KRITISCHEN ABSCHNITT ?
NEIN
FIG. 3B
A
JA
DIE HARDWARESTEUERUNGSVER-WALTUNGSEINRICHTUNG BLOCKIERT DIE ANWENDUNG,DIE EINEN ZU-GRIFF VERSUCHT.
64
WENN DIE KRITISCHE REGION FREIGEGEBEN IST,WÄHLT DIE HARDWARESTEUERUNGSVERWALTUNGS-EINRICHTUNG EINE WARTENDE AN-WENDUNG AUS,UM DEN ZUGRIFF ZU EMPFANGEN
66
B

FIG. 3A

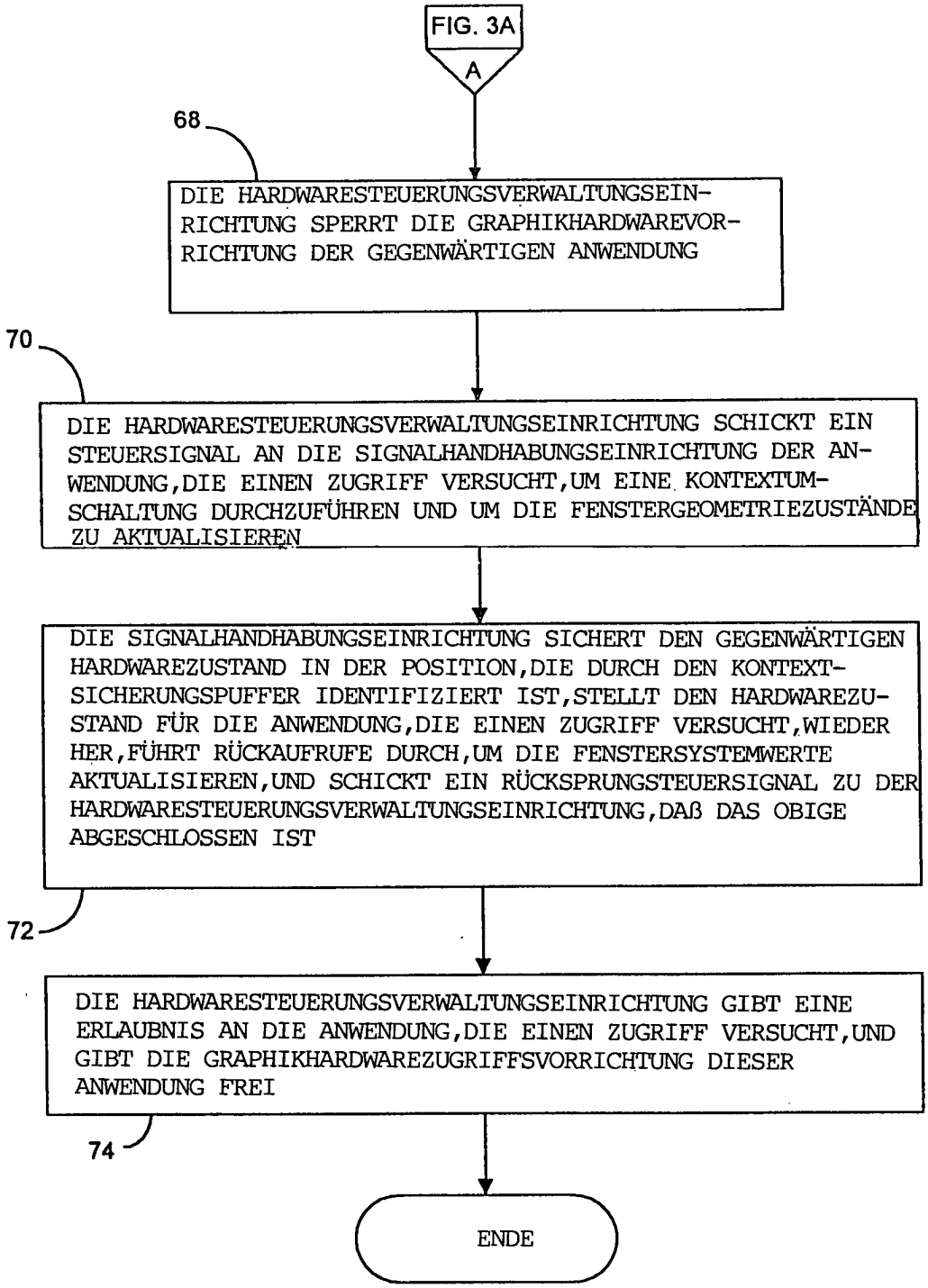


FIG. 3A
A
68
DIE HARDWARESTEUERUNGSVERWALTUNGSEINRICHTUNG SPERRT DIE GRAPHIKHARDWAREVORRICHTUNG DER GEGENWÄRTIGEN ANWENDUNG
70
DIE HARDWARESTEUERUNGSVERWALTUNGSEINRICHTUNG SCHICKT EIN STEUERSIGNAL AN DIE SIGNALHANDHABUNGSEINRICHTUNG DER ANWENDUNG,DIE EINEN ZUGRIFF VERSUCHT,UM EINE KONTEXTUMSCHALTUNG DURCHZUFÜHREN UND UM DIE FENSTERGEOMETRIEZUSTÄNDE ZU AKTUALISIEREN
DIE SIGNALHANDHABUNGSEINRICHTUNG SICHERT DEN GEGENWÄRTIGEN HARDWAREZUSTAND IN DER POSITION,DIE DURCH DEN KONTEXTSICHERUNGSPUFFER IDENTIFIZIERT IST,STELLT DEN HARDWAREZUSTAND FÜR DIE ANWENDUNG,DIE EINEN ZUGRIFF VERSUCHT,WIEDER HER,FÜHRT RÜCKAUFRUFE DURCH,UM DIE FENSTERSYSTEMWERTE AKTUALISIEREN,UND SCHICKT EIN RÜCKSPRUNGSTEUERSIGNAL ZU DER HARDWARESTEUERUNGSVERWALTUNGSEINRICHTUNG,DAß DAS OBIGE ABGESCHLOSSEN IST
72
DIE HARDWARESTEUERUNGSVERWALTUNGSEINRICHTUNG GIBT EINE ERLAUBNIS AN DIE ANWENDUNG,DIE EINEN ZUGRIFF VERSUCHT,UND GIBT DIE GRAPHIKHARDWAREZUGRIFFSVORRICHTUNG DIESER ANWENDUNG FREI
74
ENDE

FIG. 3B