



República Federativa do Brasil
Ministério da Economia
Instituto Nacional da Propriedade Industrial

(11) PI 0910188-8 B1



(22) Data do Depósito: 25/06/2009

(45) Data de Concessão: 22/10/2019

(54) Título: MÉTODO DE INICIALIZAÇÃO DE COMPUTADOR, PRODUTO DE PROGRAMA DE COMPUTADOR FABRICADO, E DISPOSITIVO ELETRÔNICO

(51) Int.Cl.: G06F 9/455.

(30) Prioridade Unionista: 02/07/2008 US 12/217,240.

(73) Titular(es): HEWLETT-PACKARD COMPANY.

(72) Inventor(es): KAUSHIK BARDE.

(86) Pedido PCT: PCT IB2009052750 de 25/06/2009

(87) Publicação PCT: WO 2010/001313 de 07/01/2010

(85) Data do Início da Fase Nacional: 29/12/2010

(57) Resumo: MÉTODO DE INICIALIZAÇÃO DE COMPUTADOR, PRODUTO DE PROGRAMA DE COMPUTADOR FABRICADO E DISPOSITIVO ELETRÔNICO A presente invenção provê técnicas que se relacionam a computadores pessoais e a dispositivos que compartilham arquiteturas similares, e, em particular, a um método e dispositivo para permitir melhor desempenho e segurança em programas hypervisor, e aplicativos relacionados, obtidos através do uso de múltiplas memórias nãovoláteis.

"MÉTODO DE INICIALIZAÇÃO DE COMPUTADOR, PRODUTO DE PROGRAMA DE COMPUTADOR FABRICADO, E DISPOSITIVO ELETRÔNICO"

Campo da Invenção

5 A presente invenção se relaciona geralmente a computadores e dispositivos que compartilham arquiteturas similares. A presente invenção, ademais, se relaciona, mais particularmente, a um método, produto de programa de computador, e dispositivo eletrônico que proporcionam
10 características de desempenho e segurança superiores em programas "Hypervisor" e programação de aplicativos relacionados.

Histórico da Invenção

Correntemente, a utilização de computadores pessoais se
15 tornou lugar comum, e ainda vem mostrando crescimento. Hardwares cada vez mais baratos vêm se tornando cada vez mais rápidos e incorporam cada vez mais capacidade e capacidade ao passar dos anos, e softwares aplicativos para lidar com novas tarefas e softwares de sistema para
20 administrar novos dispositivos continuam surgindo em profusão.

Um usuário típico encontra, a todo momento, novas versões de software que apresentam vantagens e desvantagens em relação aos antigos. Pode se dizer que com o passar do
25 tempo, os usuários contam com computadores com cada vez mais capacidades e quantidade de software.

Um problema enfrentado por um usuário típico diz respeito aos benefícios e (custo) de adotar novas versões de hardware e software (ou um aplicativo totalmente novo)
30 versus benefícios de continuar com o produto mais antigo, cuja prática domina. Os fatores a serem considerados são complexos (e além do escopo da presente invenção), e ademais soluções de software integrado trazem questões adicionais, na medida que tendem a limitar a capacidade
35 de diferentes aplicativos (ou diferentes variantes dos mesmos aplicativos) de coexistirem harmoniosamente. Um problema comum deste tipo surge se um usuário contar,

por exemplo, com um driver para um certo hardware, designado para trabalhar somente com um pré-determinado OS (Sistema Operacional), por exemplo, XP® da Microsoft®, e programa aplicativo que trabalha somente com Vista®.

5 Nenhuma solução atende plenamente, requerendo que o usuário tenha que tolerar algum tipo de compromisso.

Outro aspecto a ser considerado diz respeito ao tempo de inicialização (start-up) requerido pelo computador.

Em geral, quanto mais softwares residentes são tolerados, mais demorada se torna a inicialização. Mesmo quando múltiplos softwares (incluindo múltiplos OS) são interoperáveis e úteis (ou pelo menos co-existem harmoniosamente) e provêem desempenho suficiente, estes softwares ainda demandam um tempo de inicialização

15 excessivo em comparação com a utilização de um único OS. O tempo de inicialização de computador facilmente pode vir a se tornar um aspecto irritante a um usuário, que meramente deseja seguir com a tarefa em mãos.

Um outro problema que vem aumentando diz respeito aos chamados "malware". Vários melhoramentos que objetivam bloquear os citados "malwares" vêm sendo propostos, mas freqüentemente se baseiam em soluções integradas, ligadas a um fornecedor específico, ou pior, a uma particular versão de um software de um certo fornecedor, que torna ainda mais difícil a administração de múltiplas variantes de software. Ademais, tais soluções podem usar freqüentemente a solução "segurança por obscuridade", que provê ao usuário um reduzido controle em seu destino, e se mostra menos adequado para uso com produtos de fonte aberta (open source).

30 Sumário da Invenção

Por conseguinte, a presente invenção provê um método de inicialização de um computador para suportar máquinas virtuais. Outros dispositivos eletrônicos, incluindo computadores pessoais ou controladores digitais completamente diferentes, também podem ser usados.

Métodos inventivos podem incluir alocar regiões de

memória e carregar programas a partir de memórias não-voláteis. Tais programas podem incluir um programa hypervisor, programa para carregar hypervisor, e programas de Appliance. Ademais, heaps de memória podem
 5 ser criados para vários propósitos, e memórias não-voláteis, tais como dispositivos de disco, podem ser processados, por exemplo para carregar Sistemas Operacionais OSs.

De acordo com um aspecto, a presente invenção provê um
 10 método de inicialização de um computador pessoal para suportar máquinas virtuais, compreendendo as etapas de:

- alocar uma primeira região de memória de leitura-escrita;

- carregar a partir de uma primeira memória não-volátil para uma primeira região:
 15 um primeiro programa selecionado de uma lista, que consiste de:

- um programa hypervisor HV; e

- um programa HSC que opera para carregar o programa
 20 hypervisor na primeira região;

- criar um heap de memória na memória de leitura-escrita, excluindo a primeira região; e

- passar o controle para o primeiro programa.

De acordo com um aspecto adicional, a presente invenção
 25 provê um produto de programa de computador fabricado, compreendendo pelo menos uma mídia legível por computador incluindo instruções de computador codificadas, que, ao serem executadas, em pelo menos um computador, inicializam este pelo menos um computador para suportar
 30 máquinas virtuais através das etapas que compreendem as ações de:

- alocar uma primeira região da memória de leitura-escrita;
- carregar a partir da primeira memória não-volátil para a primeira região:

- um primeiro programa selecionado a partir de uma lista que consiste de:

- um programa hypervisor; e

- um programa que opera para carregar programa hypervisor na primeira região;
criar um heap de memória da memória de leitura-escrita, excluindo a primeira região; e
- 5 passar o controle para o primeiro programa.
- De acordo com um outro aspecto adicional, a presente invenção provê um dispositivo eletrônico compreendendo: meios para executar instruções programadas; e meios não-voláteis para memorizar instruções de
- 10 computador, que, quando são executadas pelos citados meios de execução, inicializam o citado dispositivo eletrônico para suportar máquinas virtuais através das etapas que compreendem as ações de:
- alocar uma primeira região da memória de leitura-escrita;
- 15 ~~carregar a partir da primeira memória não-volátil para a primeira região:~~
- um primeiro programa selecionado de uma lista consistindo de:
- um programa hypervisor; e
- 20 um programa que opera para carregar programa hypervisor na primeira região;
- criar um heap na memória de leitura-escrita, excluindo a primeira região; e
- passar o controle para o primeiro programa.
- 25 Desde que o programa hypervisor pode ser carregado antes ou depois de alocar e/ou particionar o heap de memória.
- Em uma configuração, o heap pode expressamente excluir a memória dedicada a HV, visando eliminar (ou reduzir substancialmente) a visibilidade do HV a partir de
- 30 programas posteriormente carregados, especialmente a partir de um sistema operacional genérico.
- Em uma configuração, depois de instalar o heap, o BIOS pode transferir o controle de volta para HSC, que então carrega e inicia em um modo de memória virtual e/ou em um
- 35 modo protegido do HV. Tipicamente, HV e/ou BIOS, em conjunto, lêem informações paramétricas de tendência da memória usadas para configuração BIOS (ou alguma outra

memória não-volátil oculta), qual informação pode ser usada para particionar o heap. Assim, a informação para guiar uma tendência é disponibilizada para HSC e HV.

Diversas variações destes aspectos serão discutidas junto com configurações alternativas exemplares. Projetos melhorados para firmware/ software provêem um compromisso superior com respeito aos problemas delineados acima.

Descrição Resumida dos Desenhos

As vantagens e aspectos acima mencionados e relacionados da presente invenção serão mais bem entendidos e apreciados a partir de uma revisão minuciosa da descrição detalhada a seguir da presente invenção, em conexão com desenhos anexos que estão incorporados e constituem parte da especificação, ilustrando uma configuração da presente invenção, onde os mesmos números de referência representam elementos iguais ou similares, nos quais:

A figura 1 é um diagrama esquemático de blocos de um dispositivo eletrônico configurado para implementar um produto de programa de computador e/ou método de inicialização de computador, de acordo com uma configuração da presente invenção;

A figura 2 é um fluxograma ilustrando as etapas realizadas pelo dispositivo eletrônico, quando se implementa uma configuração da presente invenção;

A figura 3 é um fluxograma ilustrando as etapas realizadas quando da implementação de carregamento e execução de hypervisor, appliances, e sistema operacional principal, em uma configuração exemplar da presente invenção;

A figura 4 é um fluxograma ilustrando as etapas realizadas, quando se virtualiza pedidos de informação de memória de uma máquina virtual em uma configuração exemplar da presente invenção;

A figura 5 é um fluxograma ilustrando as etapas realizadas quando se passa para um modo de facilidade reduzida em uma configuração da presente invenção;

A figura 6A mostra como uma configuração exemplar da

presente invenção pode ser codificada em uma mídia de computador; e

A figura 6B mostra como uma configuração exemplar da presente invenção pode ser codificada, transmitida, 5 recebida, e decodificada usando ondas eletromagnéticas. Para efeito de conveniência de descrição, serão dados os mesmos números de referência a componentes que forem idênticos ao longo dos vários desenhos.

Descrição Detalhada das Configurações Preferidas

10 Na descrição que se segue, para propósito de clareza e facilitar o entendimento, salienta-se que nem todos dos numerosos componentes que aparecem nos diagramas esquemáticos e/ou desenhos serão descritos. Os numerosos componentes são mostrados nos desenhos para prover 15 àqueles habilitados na técnica uma descrição abrangente e capacitante com respeito à presente invenção. A operação dos componentes deve ser plenamente entendida e aparente por aqueles habilitados na técnica habilitados na técnica.

20 Uma configuração exemplar da presente invenção será agora descrita com referência à figura 1.

A figura 1 é um diagrama esquemático de blocos de um dispositivo eletrônico, que implementa uma funcionalidade de uso de memória, de acordo com uma configuração da 25 presente invenção. Em uma configuração exemplar, o dispositivo eletrônico 10 é implementado na forma de computador pessoal, por exemplo, um computador de mesa, notebook, computador tablet, ou qualquer dispositivo de computação adequado.

30 Embora a descrição se refira à operação de um computador pessoal, aqueles habilitados na técnica deverão apreciar que o dispositivo eletrônico 10 também pode vir a ser implementado como PDA, um dispositivo de comunicação sem-fio, por exemplo, telefone celular, controladores ou 35 dispositivos incorporados, por exemplo, aparelhos acessórios separados (set-top), dispositivos de impressão, ou combinações destes, para operar e/ou

interoperar com a presente invenção.

O computador pessoal 10 pode incluir pelo menos um controlador ou processador 12, configurado para controlar a operação global do dispositivo eletrônico 10. Qual
5 processador ou controlador freqüentemente é controlado por uma CPU (de Central Processing Unit, i.e. Unidade de Processamento Central) ou MPU (de Microprocessor Unit, i.e. Unidade de Micro-processamento). O processador 12 pode incluir um ALU (Arithmetic/Logic Unit, i.e. Unidade
10 Aritmética/Lógica) para fazer operações computacionais, um ou mais registros para armazenamento temporário de dados e instruções, e um seqüenciador ou controlador para controlar operações do computador pessoal 10. Em uma configuração, o processador 12 pode incluir um micro-
15 _processador Celeron® ou Centrino® da Intel® ou Power PVC® da International Busines Machines®. Em adição, qualquer de uma variedade de outros processadores, incluindo da Sun Microsystems®, MIPS®, NEC®, Cyrix®, e outros, pode
ser usado para implementar processador 12, que, ademais,
20 não se limita a micro-processadores, podendo assumir outras formas, i.e. micro-controladores, processadores de sinal digital, hardware dedicado ASIC (Application-Specific Integrated Circuit, i.e. Circuito Integrado de Aplicação Específica), máquinas de estado, ou softwares
25 executados em um ou mais processadores distribuídos em rede.

O processador 12 pode ser acoplado a um controlador de barramento 14 por meio de um barramento de CPU 13. Qual controlador 14 pode incluir um controlador de memória
30 integrado 15, que, em uma configuração, pode ser separado do controlador de barramento 14. O controlador de memória integrado 15 pode prover uma interface para acessar, por meio de um processador 12, ou outros dispositivos, uma memória de um sistema de leitura-escrita, por exemplo,
35 memória RAM (memória de acesso randômico).

O controlador de barramento 14 pode ser acoplado a um barramento de sistema 18, por exemplo, um barramento PCI

(Peripheral Component Interconnect, i.e. Interconexão de Componente Periférico). O barramento de sistema 18 pode ser acoplado um controlador de dispositivo periférico 24 também conhecido como controlador I/O (input/ output, 5 i.e. entrada/ saída). Por sua vez, o controlador de dispositivo periférico 24 pode ser acoplado a vários outros dispositivos 25, 26, 27, tal como, por exemplo, teclado, mouse, impressora etc... Outros componentes (não mostrados na figura 1) provavelmente também poderiam 10 estar presentes, por exemplo, um controlador de tela acoplado à tela, e um controlador de comunicação de rede, são muito comuns.

O barramento de sistema 18 também pode ser acoplado a um número de memórias não-voláteis, para efeito de exemplo, 15 na prática, podem haver mais ou menos memórias não-voláteis. Em outras configurações, memórias podem ser acopladas de outras maneiras, tal como, por exemplo, diretamente ao controlador de barramento 14.

Como indicado na figura 1, memórias não-voláteis 20 exemplares 30, 31, 32, 33 podem prover de diferentes tecnologias e com as respectivas capacidades. Como mostrado na figura 1, memória não-volátil 30 pode ser implementada como uma memória apenas de leitura ROM e incluir um POST 40 (Power-On Self-Test, i.e. Auto-Teste 25 de Ligação) e CSS 41 (Core System Software, i.e. Software de Sistema de Núcleo), assim como outros componentes, tal como DDs (Device Drivers, i.e. Drivers (Controladores) de Dispositivos).

Ainda se referindo à figura 1, as terceira e quarta 30 memórias não-voláteis 32, 33 podem incluir vários componentes. Tipicamente, uma das memórias não-voláteis pode ser implementada como memória Flash ou SDM (Secure Digital Memory, i.e. Memória Digital Segura) que tipicamente compreende Firmware BIOS (Basic Input/Output 35 System, i.e. Sistema Básico de Entrada/ Saída) 42. A memória Flash é usualmente incorporada como um tipo de EEPROM (Electrically Erasable Programmable Read-Only

Memory, i.e. Memória Eletricamente Apagável Apenas de Leitura) de bloco orientado.

Outros componentes de firmware e software, por exemplo, Hyperspace® da Phoenix Technologies® Ltd, podem ser armazenados em qualquer memória não-volátil, por exemplo, a terceira memória não-volátil 32 podem incluir uma cópia de programa Hypervisor 61 (também conhecido como VMM (Virtual Machine Monitor, i.e. Monitor de Máquina Virtual)) e/ou programas aplicativo relativamente pequenos, os chamados Appliances 62 (ou Applets).

Embora firmware e software possam ser armazenados em memórias não-voláteis, firmware e software são comumente transferidos para a memória do sistema 16 antes de sua execução através de dispositivo driver de bloco-orientado ou técnica de memória sombra, como conhecido na técnica. Excepcionalmente, instruções, especialmente instruções POST usadas no setup dos primeiros computadores, podem ser executadas diretamente fora do ROM, ou etc. tal como instruções usadas para implementar a própria operação de memória sombra (shadow-memory).

Em combinação, e em muitas permutações, as memórias não-voláteis da figura 1 podem incluir instruções (de firmware ou software) para executar as etapas ou ações descritas abaixo, com referência a outras figuras.

A figura 2 é um fluxograma ilustrando as etapas realizadas pelo dispositivo eletrônico, quando da implementação de uma configuração da presente invenção, a partir da condição de ligação (power on) 200. A figura 2 é meramente exemplar, e muitas configurações alternativas serão possíveis dentro do escopo da presente invenção. Ademais, a figura 2 é de um nível um tanto alto, e detalhes adicionais serão descritos mais adiante em conexão com as figuras.

Na etapa 210, o hardware é inicializado como ação preliminar para ligar o hardware, de modo a conseguir outras capacidades. Então, se inicia POST 220, tendo muitas características não mostradas na figura 2.

Em algum ponto no processo POST, há oportunidade 230 para intervenção humana 240, para reconfiguração manual de parâmetros de memória não-volátil, tal como mini SDM (Secure Digital Memory, i.e. Memória Digital Segura),
5 seguido de um retorno ao começo de uma sequência de inicialização de ligação 210 (Power Up).

Na etapa 250, é verificado se são solicitados serviços Hypervisor, como da Phoenix Technologies® Hypercore™,. Tal pedido pode ser um serviço de boot, tal como, em
10 resposta a um parâmetro de configuração BIOS. A determinação feita assim (etapa 260), se serviços Hypervisor não estiverem sendo pedidos, então, na etapa 270, um ajuste de memória convencional é realizado, e na etapa 280, como um sistema operacional é carregado por
15 boot (bootload) na etapa 299, o controle é consentido a partir do BIOS (neste exemplo, para o Sistema Operacional).

Na etapa 300, serviços Hypervisor foram solicitados, e setup Hypervisor reativo e Hypervisor de carregamento por
20 boot (bootload) serão discutidos abaixo, com referência à figura 3. Depois do que o setup é completado.

A figura 3 mostra setup de carregamento por boot HV e ações associadas, de acordo com uma configuração da presente invenção, e começando 300 com o início do setup
25 Hypercore. Tipicamente, nesta fase de execução, os códigos de instrução fazem parte do BIOS (ou de outro firmware) e podem ser encobertos (ocultos) na memória.

Na etapa 303, um espaço é alocado em uma partição SMM (de System Management Memory, i.e. Memória de
30 Administração de Sistema) para dados de segurança Hypervisor HV. A memória SMM é acessada em um modo de CPU, temporalmente menos usado - Modo de Administração de Sistema - destinado a serviços de nível baixo, tal como monitoramento de temperatura de CPU.

35 A seguir, na etapa 305, um espaço é alocado à HV, (Hypervisor) por exemplo PVMM (de Phoenix Virtual Machine Monitor, i.e. Monitor de Máquina Virtual Phoenix)

a partir de uma memória livre. No exemplo mostrado, esta alocação pode ser feita a partir de espaço RAM TSEG (de Top of Memory Segment - Topo do Segmento de Memória). Este espaço recém-alocado é conhecido como PVMMS (de
5 Phoenix Virtual Machine Monitor segment, i.e. segmento de Monitor de Máquina Virtual Phoenix).

Na etapa 310, HSC (Hypervisor Starter Code, i.e. Código de Partida de Hypervisor) é carregado a partir de uma memória não-volátil, por exemplo, memória Flash para
10 PVMMS. Técnicas de relocação de memória são bem conhecidas na técnica e incluem aplicação de endereços de base ajustada a deslocamentos (offsets) e técnicas similares para tornar o código independente dos endereços de memória absoluta. Em uma configuração, HSC opera
15 principalmente em Modo-Big Real (também conhecido como Modo Real Flat ou Unreal) na Intel® X86 e arquiteturas CPU little endian, e similares.

Relocation de código também é conhecido como Loading, em algumas implementações, especialmente em ambientes nos
20 quais Unix® argot é favorecido. Loading é um pouco mais amplo que Relocation, por não incluir necessariamente ajustes de endereço - ainda que tais ajustes sejam feitos em uma alta proporção de casos.

Em uma configuração, o HSC então pode operar (etapa 315)
25 para carregar um programa hypervisor. Algumas vezes, HV não é carregado diretamente, mas, ao invés, carrega Carregador (Loader), que, por sua vez, carrega HV, tal como em um arranjo de carregamento em cascata.

Programas HV são comuns na técnica, e são usados para
30 criar e hospedar ambientes de máquinas virtuais VM, nos quais programas, substancialmente capazes de serem executados nus (i.e. sem HV) no computador, são recebidos pelo HV. Máquinas virtuais são bem conhecidas, e se tratam de mecanismos que impedem a visibilidade entre
35 situações de VMs, de modo que programas operando com VMs, não percebem (não têm visibilidade) que não estão sozinhos no computador entre outros programas pares.

Embora não mostrado na figura 3, o controle pode retornar do HSC para um thread BIOS, onde é alocada a memória de sistema (RAM) para necessidades adicionais de espaço, por exemplo, um espaço para ACPI (Advanced Configuration and Power Interface, i.e. Configuração Avançada e Interface de Energia).

Tipicamente uma ação tardia do código BIOS é prover o setup (figura 3, etapa 320) do heap de memória, também chamado "free store" (espaço livre). O heap tende a deixar substancialmente todo "free store" (em geral RAM), disponível para ser usado quer por um sistema operacional e programas que este carrega. Na ciência computacional, a alocação de memória dinâmica se trata da alocação de armazenamento de memória para ser usado em um programa de computador durante sua execução. Pode ser visto também como um modo de distribuir a posse de recursos de memória limitados em meio a muitas peças de dados e códigos. Em uma configuração, o heap expressamente exclui a memória dedicada para HV, eliminando, ou substancialmente reduzindo, a visibilidade do HV, a partir do último programa carregado, especialmente a partir de sistemas operacionais genéricos, tal como MS Windows® ou Linux®.

Em uma configuração, depois do setup do heap, o BIOS transfere o controle de volta para HSC, que então carrega e inicia modos de memória virtual e/ou protegida do HV. Tipicamente, HV e/ou BIOS em conjunto lêem informações de tendência paramétricas da memória, que são usadas na configuração BIOS (ou outra memória não-volátil oculta) quais informações podem ser usadas para partição do heap (etapa 325) na memória dedicada a múltiplas VMs.

Ainda referindo-se à figura 3, na etapa 330 o programa hypervisor é carregado, tipicamente pelo HSC. Em outras configurações, HV pode ser carregado antes de alocar e/ou particionar heap de memória (etapas 320, 325). Em uma implementação, HSC pode ser executado em Modo Big Real CPU, tal como modo de memória protegida ou virtual. As etapas 335 de deixar modo Real quando entra HV são

implementações específicas.

Assim, por ser capaz na etapa 340 de usar informação de partição de heap de memória para VMs, HV pode seguir fazendo setup de VMs. Na etapa 345, pequenos programas aplicativos (Appliances ou Applets) podem ser carregados a partir de uma memória não-volátil ou separados e protegidos de outra forma na memória não-volátil principal, que, tipicamente, é um disco rígido.

Com certeza, os programas carregados por HV tipicamente podem, se desejado, tratar alocação de memória como próprio heap, e adicionalmente particioná-lo para vários aplicativos, drivers de dispositivo, etc..

Neste ponto, o HV segue a seqüência de procedimento de carregamento por boot não-HV de CPU, por exemplo, na etapa 350, pode emitir 19h Interrupt, o 19h Interrupt é tomado pelo código que (etapa 355) carrega MBR a partir da memória não-volátil principal (tipicamente disco rígido) e passa o controle (etapa 360) para um programa encontrado com MBR e que pode ser, por exemplo, um carregador de boot (bootloader) no MBR ou um programa para o qual MBR inclua alguma forma de estação de indireção (apontador de disco). Na etapa 399, isto completa as ações de alocar espaço para carregar e executar HSC e HV e Appliances e/ou sistema operacional principal hospedado pelo HV.

Com certeza, a maior parte dos HVs faz muito mais que descrito aqui, como conhecido por aqueles habilitados na técnica, incluindo virtualização de pedidos I-O (Entrada-Saída) (endereçados para hardware ou drivers de sinais) e para virtualização de outros pedidos de sistema. Por exemplo, em arquiteturas x86, um HV pode acoplar 15h Interrupt, de modo que este virtualize os conhecidos pedidos de memória 15h Int/EAX= E820h, de modo que o sistema operacional veja sua própria partição de heap, e não todo heap. O BIOS tipicamente provê visibilidade a todo heap, e o HV pode atuar para impedir esta visibilidade, exceto para pedidos que se originem no

próprio HV (ou pelo menos não dentro do código carregado por HV, nem código residente no heap).

Assim, uma vez assumindo o controle, o HV pode manter controle do computador até o próximo reset de sistema, freqüentemente em resposta à intervenção por controle SMM (tipicamente usado para eventos mais significativos, tal como detecção de sobreaquecimento ou fechamento incipiente de sistema).

A figura 4 mostra um exemplo de virtualização por HV de acordo com a presente invenção. Na etapa 400, a entrada é feita em uma função de serviço, que responde a um pedido de memória, virtualizado por HV. Tal pedido de memória tipicamente é configurado como ISR (Interrupt Service Routine, i.e. Rotina de Serviço de Interrupção) para 15h interrupt software Int (etapa 410).

Na etapa 420, é verificado se o pedido é uma memória informacional do formato particular suportada (15h Int/EAX= E820h), sendo que grande parte destes formatos é conhecida na técnica. Mas, se o pedido de interrupção for de outro tipo, então, na etapa 425, o controle passa para o próximo ISR na sequência.

Assumindo que o pedido deve ser atendido, na etapa 430, se determina qual VM pede informação de memória. Se o pedido não vier de VM (etapa 440), então o pedido não será virtualizado, e, na etapa 445, se segue para outra rotina ISR na sequência (se houver).

Na etapa 450, uma resposta é compilada para o pedido 15h Int/EAX= E820h, mas virtualizada para VM específica. A partição de memória de VM é tratada como se o heap inteiro pretendesse compilar respostas virtualizadas para os pedidos de memória. Na etapa 499, isto completa rotina ISR para atender pedidos de máquina virtual.

Referindo-se agora à figura 5, onde se descreve a ação do HV quando são realizados serviços Hypercore associados à transição em operação para facilidades reduzidas, na forma racionalizada de operação. Esta capacidade é complementar, e amplamente permitida pelas facilidades

descritas acima fazendo referência a outras figuras.

Uma configuração exemplar da presente invenção poderia usar um modo de facilidade reduzida, o sistema operacional primário carregado via MBR é suspenso/
5 desabilitado, e o disco rígido associado (que é um voraz consumidor de energia) é colocado no estado desligado, mas com os programas Appliance ainda sendo executados usando uma memória não-volátil de semi-condutor. No entanto, o uso de facilidade reduzida, em condição de
10 operação de energia baixa, constitui apenas um exemplo, os dois aspectos são potencialmente separáveis.

Na etapa 500 da figura 5, inicia o processo para passar para a operação de estado de energia baixa. Na etapa 510, o usuário é notificado que o computador está passando
15 para modo Appliance, em resposta a um estímulo.

Então, na etapa 520, são enviados sinais aos sistemas operacionais OSs carregados nas VMs para avisá-los para
associar operações, que podem incluir completação de
operações I-O pendentes. Tipicamente, os OSs têm ganchos
20 para este propósito, em conexão com máquinas que passam para um estado "Dormente" (Sleep) ou similar, de acordo com escolhas de implementação de hardware e/ou software.

Na etapa 530, faz-se um pedido para desligar discos rígidos e reduzir a velocidade de clock da CPU ou o
25 número de multiprocessador CPUs ativos, ou prover outras mudanças com propósito de poupar energia, como apropriado. Tipicamente, tal controle é controlado por software que executado em SMM, e, portanto, pode ser necessário emitir SMI com um apropriado bloco de pedido
30 ao serviço requerido.

Na etapa 540, as VMs a serem suspensas são marcadas como tendo nenhum compartilhamento (ou quase nenhum compartilhamento) de tempo de CPU disponível (e intercambiadas em páginas de memória virtual, e assim por
35 adiante).

Na etapa 599, a execução do Appliance é reiniciada através de um código de saída/ retorno adequado.

Com respeito à figura 6A, as instruções de computador no dispositivo eletrônico 10 podem ser distribuídas como produtos de computadores de firmware/software fabricados 610, com uma variedade de mídias 630, sendo que as 5 instruções são gravadas com gravador de armazenamento 620. Frequentemente em produtos tão complexos, tal como aqueles que implementam a presente invenção, mais uma mídia pode ser usada para distribuição e fabricação de produtos relevantes, sendo que apenas uma mídia é 10 mostrada na figura 6A para efeito de clareza, mas, deve ser apreciado que mais que uma mídia pode ser usada, e um único produto de computador pode ser dividido por uma pluralidade de mídias.

Com respeito à figura 6B, adicionalmente, e especialmente 15 com respeito ao crescimento de utilização de Internet, produtos de computador 610 podem ser distribuídos codificando os mesmos em sinais modulados, tal como em onda. A forma de onda resultante, então, pode ser 20 transmitida por um transmissor 640, propagando em ondas portadoras eletromagnéticas moduladas 650, que são recebidas por um receptor 660. Quando de sua recepção, são demoduladas, e o sinal decodificado em uma versão adicional ou cópia do produto de computador 611, em uma memória ou dispositivo de armazenamento 11.

25 Outras topologias e outros dispositivos também poderiam ser usados para produzir configurações alternativas para a presente invenção.

REIVINDICAÇÕES

1- Método de inicialização de computador (10), para suporte de máquinas virtuais, caracterizado pelo fato de compreender as etapas de:

5 alocar (305) uma primeira região de memória de leitura-escrita (16);

 carregar (310) um programa hypervisor a partir de uma primeira memória não-volátil (30) para a primeira região:

 criar (320) um heap de memória na memória de leitura-escrita, o heap de memória excluindo a primeira região;

 particionar (325) o heap para criar uma pluralidade de partições que respondem a parâmetros conhecidos a um programa BIOS no computador, em que a pluralidade de partições são dedicadas às máquinas virtuais; e

15 fazer os resultados do particionamento visíveis ao programa hypervisor e configurar, pelo programa hypervisor, as máquinas virtuais usando a pluralidade de partições,

 em que o programa hypervisor responde a um estímulo para passar para uma operação de estado de baixo energia por:

 - suspensão da execução do sistema operacional (52) e colocando a segunda memória não volátil associada (31) em um estado desligado, e

25 - execução no programa de appliance (62) durante a operação de estado de baixo energia.

2- Método, de acordo com a reivindicação 1, caracterizado pelo fato de:

 a alocação (305) e o carregamento (310) operarem substancialmente em um modo real de CPU (Unidade de Processamento Central); e

 o programa hypervisor operar, pelo menos parcialmente, em um modo protegido ou em um modo de endereço virtual.

35 3- Método, de acordo com a reivindicação 1 ou 2, caracterizado pelo fato de

adicionalmente compreender:

fazer a primeira região invisível a pelo menos alguns programas carregados no computador depois de o primeiro programa, mas antes de ocorrer uma condição de reset ainda
5 mais tardio no computador.

4- Método, de acordo com qualquer uma das reivindicações de 1 a 3, caracterizado pelo fato de adicionalmente compreender:

fazer a primeira região invisível a consultas de
10 locação de memória a partir de instruções localizadas no heap.

5- Método, de acordo com a reivindicação 1, caracterizado pelo fato de o programa hypervisor:

alocar à primeira máquina virtual uma partição
15 selecionada a partir da pluralidade de partições; e

virtualizar um pedido a partir da primeira máquina virtual para prover informação com respeito à disponibilidade de memória de leitura-escrita.

6- Método, de acordo com qualquer uma das reivindicações de 1 a 5, caracterizado pelo fato de adicionalmente
20 compreender, antes de carregar o programa hypervisor:

uma ação de demodular um sinal a partir de uma onda eletromagnética, o sinal incluindo instruções de computador codificadas, que, quando executadas pelo
25 computador, inicializam o computador através das etapas definidas na reivindicação 1.

7- Método, de acordo com a reivindicação 6, caracterizado pelo fato de adicionalmente compreender, antes da ação de demodular:

30 uma ação de modular o sinal na onda eletromagnética impressa em uma mídia tangível.

8- Produto de programa de computador fabricado, caracterizado pelo fato de compreender:

pelo menos uma mídia legível por computador incluindo
35 instruções de computador codificadas na mesma, que, quando executadas por pelo menos um computador, inicializam o

pelo menos um computador para suportar máquinas virtuais através das etapas compreendendo as ações de:

- alocar (305) a primeira região da memória de leitura-escrita (16);
 - 5 carregar (310) um programa hypervisor a partir de uma primeira memória não-volátil (30) para a primeira região;
 - criar (320) um heap de memória da memória de leitura-escrita, o heap de memória excluindo a
 - 10 primeira região;
 - particionar (325) o heap para criar uma pluralidade de partições que respondem a parâmetros conhecidos a um programa BIOS no computador, em que a pluralidade de partições são dedicadas às máquinas virtuais; e
 - 15 fazer os resultados do particionamento visíveis ao programa hypervisor e configurar, pelo programa hypervisor, as máquinas virtuais usando a pluralidade de partições,
 - em que o programa hypervisor responde a um estímulo
 - 20 para passar para uma operação de estado de baixo energia por:
 - suspensão da execução do sistema operacional (52) e colocando a segunda memória não volátil associada (31) em um estado desligado, e
 - 25 - execução no programa de appliance (62) durante a operação de estado de baixo energia.
- 9- Dispositivo eletrônico (10), caracterizado pelo fato de compreender:
 - meios (12) para executar instruções programadas; e
 - 30 meios não-voláteis (30, 31, 32, 33) para memorizar instruções de computador, quais instruções, quando executadas pelos citados meios de execução, inicializam o citado dispositivo eletrônico para suportar máquinas virtuais através das etapas que compreendem as ações de:
 - 35 alocar (305) uma primeira região da memória de leitura-escrita (16);

carregar (310) um programa hypervisor a partir de uma primeira memória não-volátil (30) para a primeira região:

criar (320) um heap na memória de leitura-escrita (16), o heap na memória excluindo a primeira região; particionar (325) o heap para criar uma pluralidade de partições que respondem a parâmetros conhecidos a um programa BIOS no dispositivo eletrônico, em que a pluralidade de partições são dedicadas às máquinas virtuais; e

fazer os resultados do particionamento visíveis ao programa hypervisor e configurar, pelo programa hypervisor, as máquinas virtuais usando a pluralidade de partições,

em que o programa hypervisor responde a um estímulo para passar para uma operação de estado de baixo energia por:

- suspensão da execução do sistema operacional (52) e colocando a segunda memória não volátil associada (31) em um estado desligado, e

- execução no programa de appliance (62) durante a operação de estado de baixo energia.

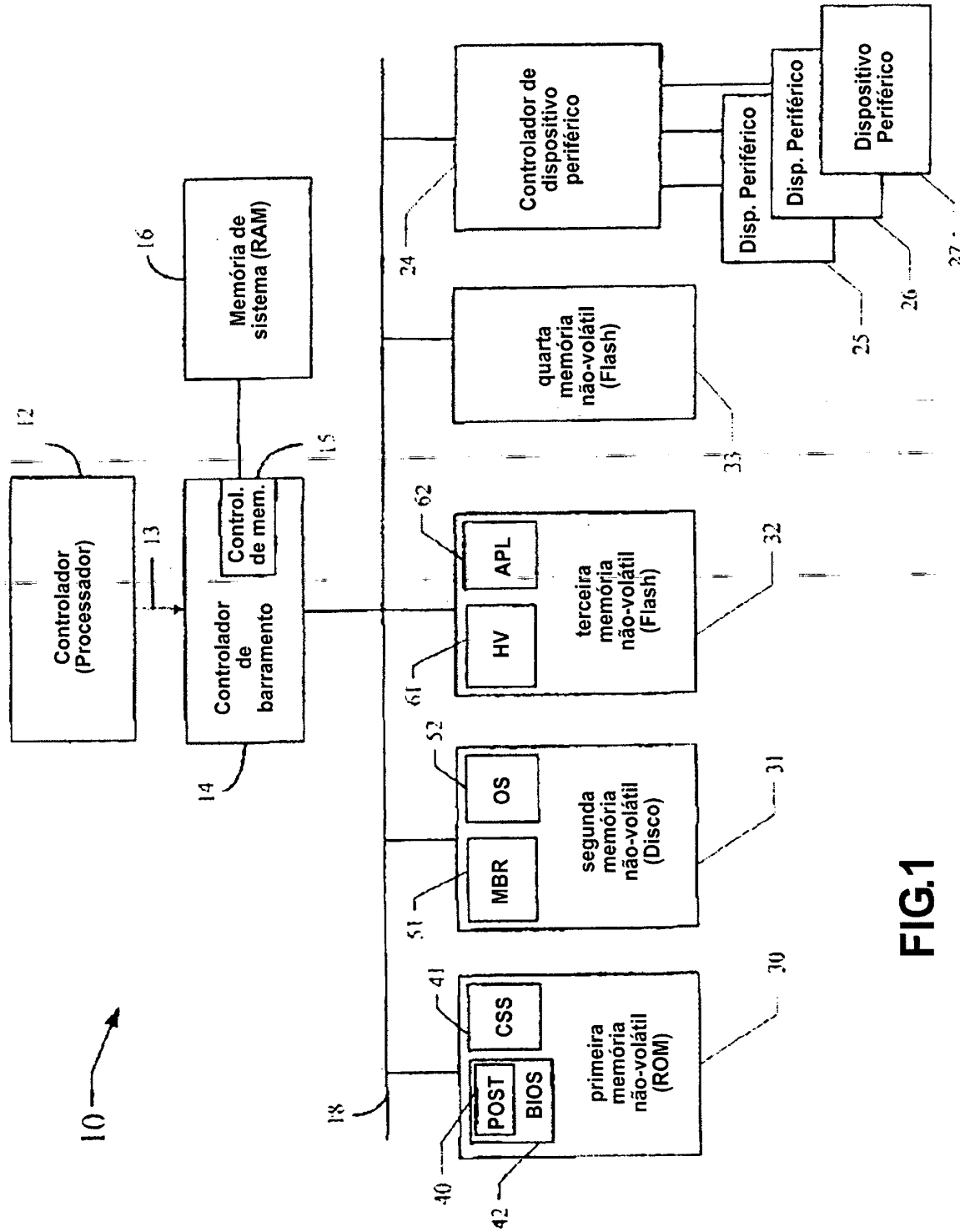


FIG.1

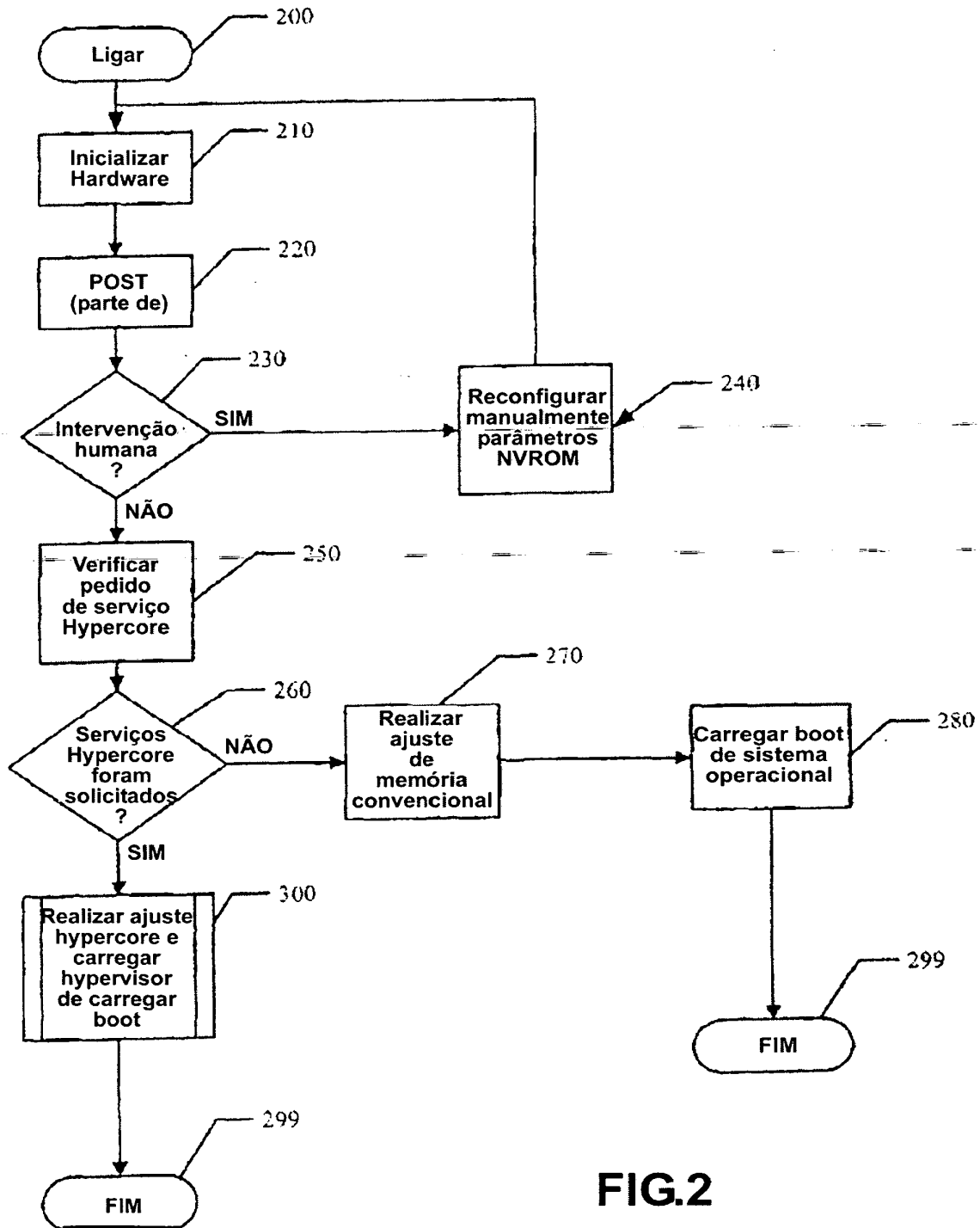


FIG.2

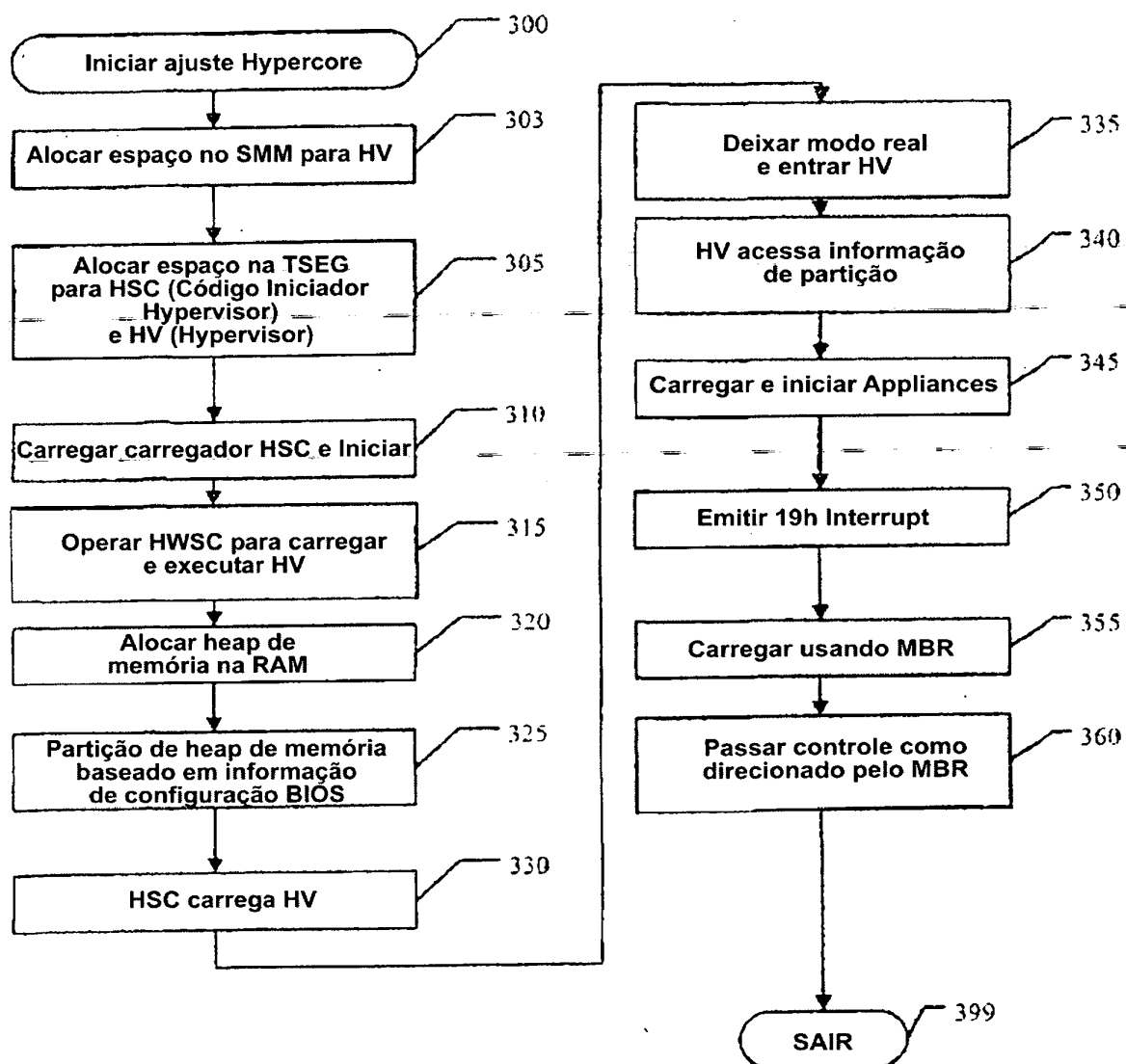


FIG.3

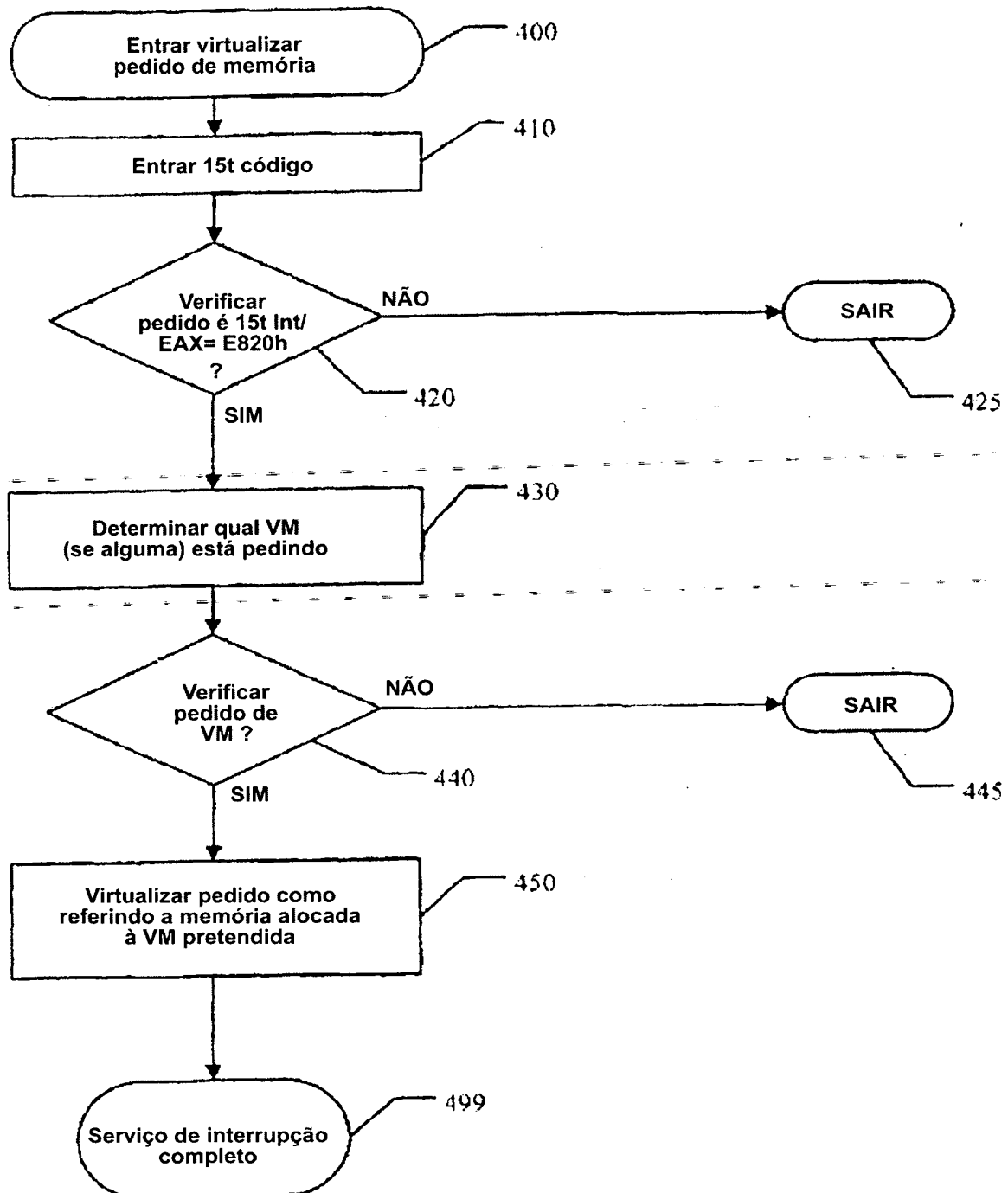


FIG.4

5/6

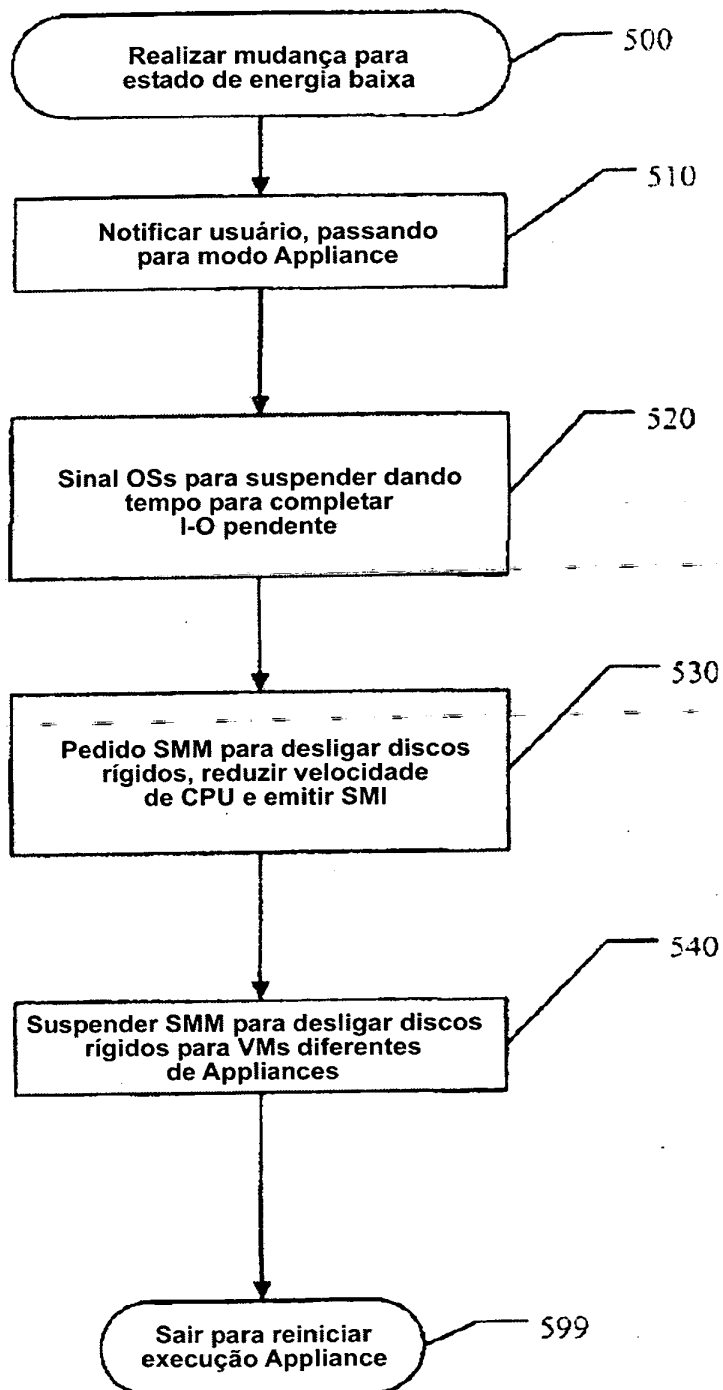


FIG.5

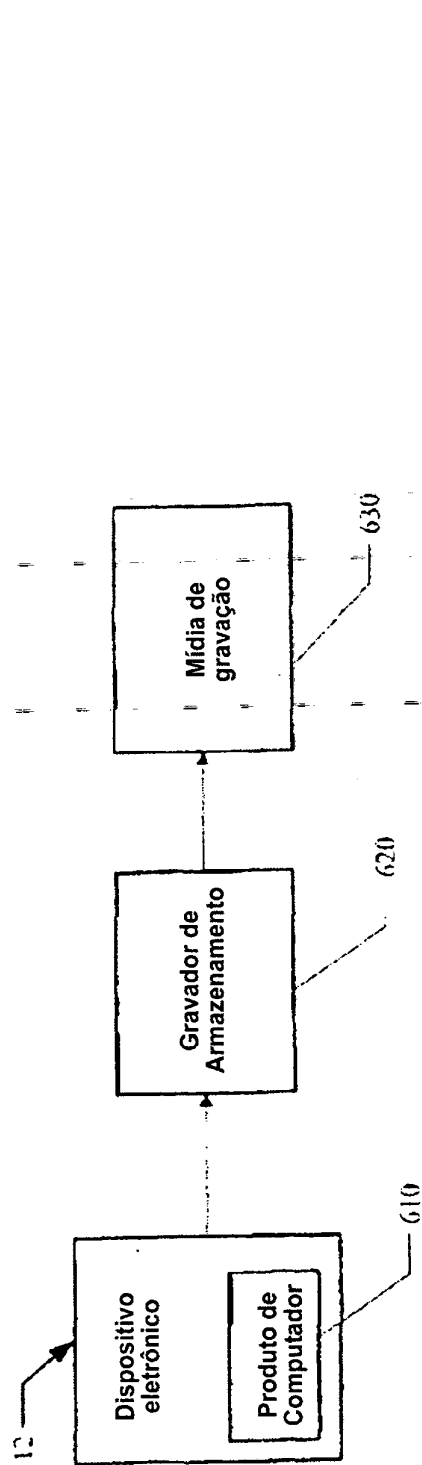


FIG. 6A

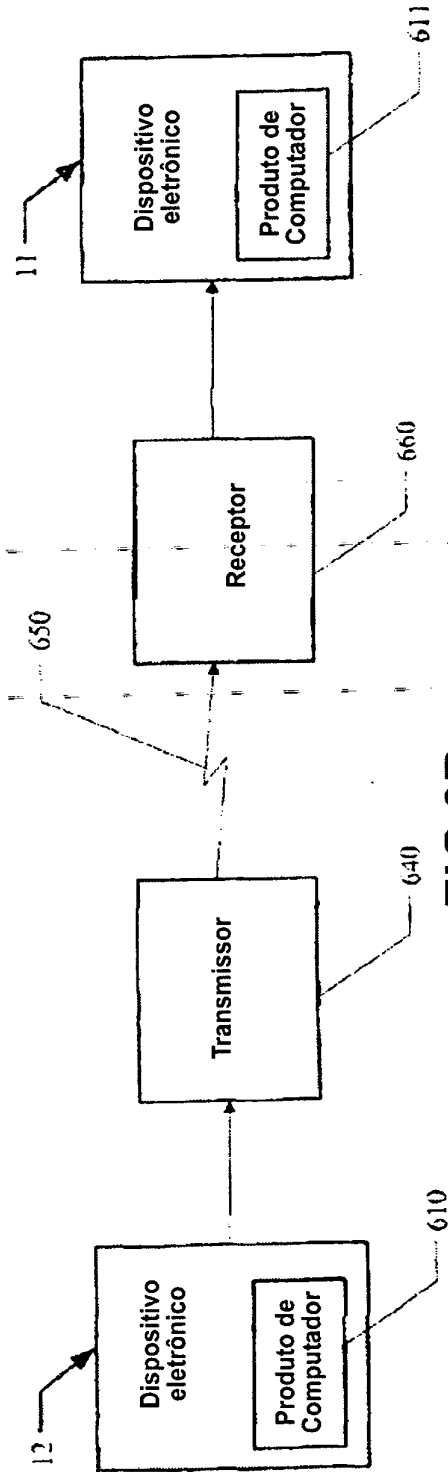


FIG. 6B