(12) PATENT
(19) AUSTRALIAN PATENT OFFICE

(11) Application No. AU 199875084 B2
(10) Patent No. 774946

(54) Title
Video game system

(51)6 International Patent Classification(s)
G09B 009/28 G06F 003/033
A63F 009/22 G06F 019/00
B06B 001/02

(21) Application No: 199875084 (22) Application Date: 1998.07.09

(30) Priority Data

(31) Number	(32) Date	(33) Country
9-192166	1997.07.17	JP

(43) Publication Date : 1999.01.28
(43) Publication Journal Date : 1999.01.28
(44) Accepted Journal Date : 2004.07.15

(71) Applicant(s)
Nintendo Co., Ltd.

(72) Inventor(s)
Satoshi Nishiumi; Kazuo Koshima; Keizo Ohta

(74) Agent/Attorney
Davies Collison Cave,Level 15,1 Nicholson Street,MELBOURNE VIC 3000

(56) Related Art
AU 744892

ABSTRACT OF THE DISCLOSURE

A video game system includes a video game machine, a memory medium and a controller. A CPU included in the video game machine detects a vibration generating condition that a player object comes into collision or contact with an enemy object or a stationary object. If the vibration generating condition is detected, the CPU drives a vibration source included in the controller, thereby generating vibration on the controller.

AUSTRALIA

PATENTS ACT 1990

COMPLETE SPECIFICATION

NAME OF APPLICANT(S):

Nintendo Co., Ltd.

ADDRESS FOR SERVICE:

DAVIES COLLISON CAVE
Patent Attorneys
1 Little Collins Street, Melbourne, 3000.

INVENTION TITLE:

Video game system

The following statement is a full description of this invention, including the best method of performing it known to me/us:–

BACKGROUND OF THE INVENTION

Field of the invention

This invention relates to video game systems. More specifically, this invention relates to a novel video game system having a vibration source provided within a controller to supply operating signals to a video game machine, generating vibrations in response to game images.

Description of the prior art

The conventional business-use video game machines involves so-called a bodily-sensible game machine already placed in practical use, which is adapted to cause vibration on a player's seat or tilt in the seat in order to enhance interest in a game further more.

Meanwhile, there is a controller, disclosed, e.g. in Japanese Utility Model Laying-Open No. S62-194389 laid open on December 10, 1987, which has an electromagnet arranged within a joystick lever to cause vibration on the joystick lever due to the actuation of the electromagnet. It may be considered that such a game is available that vibration is transmitted to a player's hand by connecting a controller generative of vibration to a home-use video game machine.

In the prior art, however, there is no disclosure as to interrelations between the game images displayed on a display and vibration caused therefor. It is therefore difficult for such a prior art to utilize for a home-use video game system. Further, the above-mentioned bodily-sensible game machine is to generate vibration on the seat, and cannot be utilized for a home-use video game system without having such a seat.

SUMMARY OF THE INVENTION

A primary object of this invention is to provide a video game system which is capable of transmitting vibration to a player in a manner related to game images, or at least to provide a useful alternative.

According to the present invention there is provided a video game system, having at least a video game machine, a memory medium and a controller, wherein said video game machine includes a processing means for generating game images to be displayed on a display depending upon operation data output from said controller and according to a game program memorized in said memory medium;

said controller comprising:

a housing for being grasped by a hand of a player;

a plurality of operating switches that the player uses to generate said operation data to designate a movement or motion of a player object;

a vibration source for applying vibration to said housing; and

a drive circuit for driving said vibration source in response to a command signal supplied from said video game machine;

said memory medium comprising the following programs to be executed by said processing means:

a player object image generating program for generating an image of the player object that is varied in response to at least one operation data input from at least one of said operating switches;

another object image generating program for generating an image of at least one other object that is present around the player object;

a vibration generating condition detecting program for detecting a first vibration generating condition for generating weak vibration depending upon a display state relationship between the player object image and at least one other object image, and for detecting a second vibration generating condition for generating intense vibration depending upon said display state relationship; and

a vibration control program for generating control data to turn said vibration source on or off for each display frame of said display in response to detection of one of the first and second vibration generating conditions, according to the vibration generating

COMS ID No: SBMI-00772060 Received by IP Australia: Time (H:m) 14:54 Date (Y-M-d) 2004-05-31

condition detecting program, and causing said vibration source to generate the comparatively intense vibration or the comparatively weak vibration depending on the number of frames for which said vibration source is turned on and off;

said processing means controlling said vibration source to generate the comparatively weak vibration in response to detection of the first vibration generating condition and to generate the comparatively intense vibration in response to detection of the second vibration generating condition, according to said control data.

The present invention also provides a video game system, having at least a video game machine and a controller, wherein said controller comprises a housing for being grasped by a hand of a player, a plurality of operating switches that the player uses to generate operation data to designate a movement or motion of a player object, a vibration source for applying vibration to said housing, and a drive circuit for driving said vibration source in response to a command signal supplied from said video game machine,

said video game machine, comprising:

a player object image generating means for generating an image of the player object that is varied in response to at least one operation data input from at least one of said operating switches;

another object image generating means for generating an image of at least one other object that is present around the player object;

a vibration generating condition detecting means for detecting a first vibration generating condition for generating weak vibration depending upon a display state relationship between the player object image and at least one other object image, and for detecting a second vibration generating condition for generating intense vibration depending upon said display state relationship; and

a vibration generating means for causing through said drive circuit said vibration source to generate vibration at timing synchronous with the generation of each display frame on a display, and for generating control data to turn said vibration source on or off for each display frame of said display in response to detection of one of the first and second vibration generating conditions by the vibration generating condition detecting means, and causing said vibration source to generate the comparatively intense vibration or the comparatively weak vibration depending on the number of frames for which said

COMS ID No: SBMI-00772060 Received by IP Australia: Time (H:m) 14:54 Date (Y-M-d) 2004-05-31

vibration source is turned on and off;

said vibration generating means including a drive means for controlling said vibration source to generate the comparatively weak vibration in response to detection of the first vibration generating condition and to generate the comparatively intense vibration in response to detection of the second vibration generating condition, according to said control data.

BRIEF DESCRIPTION OF THE DRAWINGS

Preferred embodiments of the present invention are herein described, by way of example only, with reference to the accompanying drawings, wherein:

Figure 1 is a schematic illustrative view showing a video game system according to one embodiment of this invention;

Figure 2 is a block diagram showing in detail the video game machine in Figure 1;

Figure 3 is a block diagram showing in detail a controller control circuit in Figure 2;

Figure 4 is a block diagram showing in detail the controller control circuit and vibration cartridge in Figure 2;

Figure 5 is an illustrative view showing in detail the vibration cartridge;

Figure 6 is a circuit diagram showing a vibration generating circuit included in the vibration cartridge;

Figure 7 is an illustrative view showing a memory map in an external ROM;

Figure 8 is an illustrative view showing a memory map in a RAM;

Figure 9 is a flowchart showing an overall operation in the Figure 1 embodiment;

Figure 10 is a flowchart showing a vibration generating condition detecting subroutine;

Figure 11 is a flowchart showing a hit determining subroutine;

Figure 12 is a flowchart showing a vibration generating subroutine;

Figure 13 is an illustrative view showing collision of a player object against an enemy object;

Figure 14 is an illustrative view showing collision of the player object against a stationary object;

Figure 15 is a graph showing a vibration pattern to generate intense and brief

COMS ID No: SBMI-00772060 Received by IP Australia: Time (H:m) 14:54 Date (Y-M-d) 2004-05-31

vibration;

Figure 16 is a graph showing a vibration pattern to generate weak and long vibration; and

Figure 17 is an illustrative view showing a memory map in the RAM of an embodiment in this invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

Referring to Figure 1, a video game system in this embodiment includes a video game machine 10, a ROM cartridge 20 as one example of a memory medium, a display 30 connected to the video game machine 10, and a controller 40. The controller 40 is detachably mounted with a vibration cartridge 50.

The controller 40 is structured by a plurality of switches or buttons provided on the housing 41 formed graspable by both or one hand. Specifically, the controller 40 includes handles 41L, 41C, 41R downwardly extending respectively from a left end, a right end and a center of the housing 41, providing an operating area in an upper surface of the housing 41. In the operating area, there are provided an analog-inputtable joystick (hereinafter referred to as "analog joystick") 45 at a central lower portion thereof, a cross-shaped digital direction switch (hereinafter called "cross switch") 46 on the left side, and a plurality of button switches 47A, 47B, 47D, 47E and 47F on the right side.

The analog joystick 45 is used to input a moving direction and/or moving speed or moving amount of a player object (an object operable by a player through the controller 40) determined by the amount and direction of inclination of the joystick. The cross switch 46 is used to designate a moving direction of the player object, in place of the joystick 45. The button switches 47A and 47B are used to designate a motion of the player object. Button switches 47C - 47D are used to switch over a visual point of a

three-dimensional image camera or adjust speed or the like of the player object. A start switch 47S is provided nearly at a center of the operating area. This start switch 47S is operated when starting a game. A switch 47Z is provided at a backside of the central handle 41C. This switch 47Z is utilized, for example, as a trigger switch in a shoot game. Switches 47L and 47R are provided on a side surface at upper left and right of the housing 41.

Incidentally, the above-stated button switches 47C - 47F can also be used to control the moving speed (e.g. acceleration or deceleration) of the player object in a shoot or action game, besides for the purpose of switching the camera visual point. However, these switches 47A - 47F, 47S, 47Z, 47L and 47R can be arbitrarily defined in their function depending upon a game program.

Figure 2 is a block diagram of the video game system of the Figure 1 embodiment. The video game machine 10 incorporates therein a central processing unit (hereinafter referred to as "CPU") 11 and a coprocessor (reality coprocessor: hereinafter referred to as "RCP") 12. The RCP 12 includes a bus control circuit 121 for controlling buses, a signal processor (reality signal processor; hereinafter referred to as "RSP") 122 for performing polygon coordinate transformation, shading treatment and so on, and a rendering processor (reality display processor; hereinafter referred to as "RDP") 46 for rasterizing polygon data into an image to be displayed and converting the same into a data form (dot data) memorable on a frame memory.

The RCP 12 is connected to a cartridge connector 13 for unloadably loading a ROM cartridge 20 having an external ROM 21 incorporated therein, a disc-drive connector 197 for detachably mounting a disc drive 29, and a RAM 14. Also, the RCP 12 is connected with DAC (Digital/Analog Converters) 15 and 16 for respectively outputting a sound signal and video signal to be processed by the CPU 11. Further, the RCP 12 is

connected with a controller control circuit 17 to serially transfer operating data on one or a plurality of controllers 40 and/or data of the vibration cartridge 50.

The bus control circuit 121 included in the RCP 12 performs parallel/serial conversion on a command supplied in a parallel signal from the CPU via a bus, to thereby supply a serial signal to the controller control circuit 18. Also, the bus control circuit 121 converts a serial signal inputted from the controller control circuit 17 into a parallel signal, giving an output to the CPU 11 via the bus. The data representative of an operating state (operating signal or operating data) read out of the controller 40A - 40D is processed by the CPU 11, and temporarily stored within a RAM 14, and so on. In other words, the RAM 15 includes a storage site for temporarily memorizing the data to be processed by the CPU 11, so that it is utilized for smoothly reading and writing data through the bus control circuit 121.

The sound DAC 15 is connected with a connector 195 provided at a rear face of the video game machine 10. The image DAC 16 is connected with a connector 196 provided at the rear face of the video game machine 10. The connector 195 is connected with a speaker 31 of a display 30, while the connector 196 is connected with a display 30 such as a TV receiver or CRT.

The controller control circuit 17 is connected with a controller connector provided at the front face of the video game machine 10. The connector 18 is disconnectably connected by a controller 40 through a connecting jack. The connection of the controller 40 to the connector 18 places the controller in electrical connection to the video game machine 10, thereby enabling transmission/reception or transfer of data therebetween.

The controller control circuit 17 is used to transmit and receive data in serial between the RCP 12 and the connector 18. The controller control circuit 17 includes, as shown in Figure 3, a data transfer control circuit 171, a transmitting circuit 172, a

receiving circuit 173 and a RAM 174 for temporarily memorizing tansmission and reception data. The data transfer control circuit 171 includes a parallel/serial converting circuit and a serial/parallel converting circuit in order to convert a data format during data transfer, and further performs write/read control on the RAM 174. The serial/parallel converting circuit converts the serial data supplied from the RCP 12 into parallel data, supplying it to the RAM 174 or the transmitting circuit 172. The parallel/serial converting circuit converts the parallel data supplied from the RAM 174 or the receiving circuit 173 into serial data, to supply it to the RCP 12. The transmitting circuit 172 converts the command for reading signals from the controller 40 and the writing data (parallel data) to the vibration cartridge 50, into serial data to be delivered to channels CH1 - CH4 corresponding to the respective controllers 40. The receiving circuit 173 receives, in serial data, operational state data of the controllers inputted through corresponding channels CH1 - CH4 and data read from the vibration cartridge 50, to convert them into parallel data to be delivered to the data transfer control circuit 171. The data transfer control circuit 171 writes into the RAM 174 data transferred from the RCP 12, data of the controller received by the receiving circuit 183, or data read out of the RAM cartridge 50, and reads data out of the RAM 174 based on a command from the RCP 12 so as to transfer it to the RCP 12.

The RAM 174, though not shown, includes memory sites for the respective channels CH1 - CH4. Each of the memory sites is stored with a command for the channel, transmitting data and/or reception data.

Figure 4 is a detailed circuit diagram of the controller 40 and the vibration cartridge 50. The housing of the controller 40 incorporates an operating signal processing circuit 44, etc. in order to detect an operating state of the joystick 45, switches 46, 47, etc. and transfer the detected data to the controller control circuit 17. The operating signal

processing circuit 44 includes a receiving circuit 441, a control circuit 442, a switch signal detecting circuit 443, a counter circuit 444, a joyport control circuit 446, a reset circuit 447 and a NOR gate 448. The receiving circuit 441 converts a serial signal, such as a control signal transmitted from the controller control circuit 17 or writing data to the vibration cartridge 50, into a parallel signal to supply it to the control circuit 442. The control circuit 442 generates a reset signal to reset (0), through the NOR gate 448, count values of an X-axis counter 444X and a Y-axis counter 444Y within the counter 444, when the control signal transmitted from the controller control circuit 17 is a signal for resetting X, Y coordinates of the joystick 45.

The joystick 45 includes X-axis and Y-axis photo-interrupters in order to decompose a lever inclination into X-axis and Y-axis components, generating pulses in number proportional to the inclination. The pulse signals are respectively supplied to the counter 444X and the counter 444Y. The counter 444X counts a number of pulses generated in response to an inclination amount when the joystick 45 is inclined in the X-axis direction. The counter 444Y counts a number of pulses generated responsive to an inclination amount when the joystick 45 is inclined in the Y-axis direction. Accordingly, the resultant X-axis and Y-axis vector determined by the count values of the counters 444X and 444Y serves to determine a moving direction and a coordinate position of the player object or hero character or a cursor. Incidentally, the counter 444X and the 444Y are reset, when a reset signal is supplied from the reset signal generating circuit 447 upon turning on the power or a reset signal is supplied from the switch signal detecting circuit 443 by simultaneous depression of predetermined two switches.

The switch signal detecting circuit 443 responds to a switch-state output command supplied at an interval of a constant period (e.g. a 1/30 second interval as a TV frame period) from the control circuit 442, to read a signal varying depending upon a depression

state of the cross switch 46 and the switches 47A - 47Z. The read signal is delivered to the control circuit 442. The control circuit 442 responds to a read-out command signal of operational state data from the controller control circuit 17 to supply in a predetermined data format the operational state data on the switches 47A - 47Z and count values of the counters 444X and 444Y to the transmitting circuit 445. The transmitting circuit 445 converts the parallel signal outputted from the control circuit 442 into a serial signal, and transfer it to the controller control circuit 17 via a converting circuit 43 and a signal line 42. The control circuit 442 is connected with a joystick control circuit 446 via an address bus and a data bus as well as a port connector 46. The joyport control circuit 446 performs data input/output (or transmission/reception) control according to a command from the CPU 11 when the vibration cartridge 50 is connected to the port connector 46.

The vibration cartridge 50 is structured by connecting the RAM 51 to the address bus and data bus and connecting the RAM 51 with a battery 52. The RAM 51 is a RAM having a capacity (e.g. 256k bits), for example, of lower than a half of a maximum memory capacity accessible through the address bus. The RAM 51 is to store backup data in relation to a game, and keeps backup data by the application of electric power from the battery 52 even if the vibration cartridge 50 is withdrawn from the port connector 449. This vibration cartridge 50 incorporates a vibration generating circuit 53 therein.

Now the vibration cartridge will be explained in detail with reference to Figure 5. The vibration cartridge 50 includes a case 501 and a back lid 502 attached to the case 501. The vibration cartridge 50, formed by this case 501 and the back lid 502, is dismountably mounted in an opening portion of the controller 40 shown in Figure 1.

The case 501 has a substrate 503 accommodated therein. On the substrate 503 are mounted, besides the afore-mentioned RAM 51 and a backup battery 52, a battery 504

and a driving circuit 505 that constitute the vibration generating circuit 53 of Figure 4. Incidentally, the substrate 503 has a plurality of terminals 506 at an edge portion toward this so that the terminals 506 are to be connected to a connector (not shown) formed at an opening portion of the above-stated controller 40. Through these terminals 506 are received data and addresses supplied from the CPU 11 (Figure 2), that is, the controller control circuit 17 of the video game machine 10.

The back lid 502 is fixed with a vibration source 507 constituting the vibration generating circuit 53. In this embodiment, the vibration source 507 employs a motor to generate vibrations. However, it is of course possible to utilize, besides a motor, other devices, such as a solenoid, for generating vibration by the application of power. Incidentally, "FM16", "FM23", "FM25", "FM29" or "CM-5", etc. made by Tokyo Parts Industry Co., Ltd. are available as a vibration generating motor. Where using an "FM" motor, an eccentric member is attached to a rotary shaft built in a cylindrical case. When the rotary shaft is rotated, the eccentric member is rotated to cause vibration on the case. When a "CM" motor is used, an armature coil is eccentrically arranged. By rotating the armature, vibration is caused. Incidentally, if a solenoid is used, vibration occurs due to reciprocal movement of a magnetic core provided within the solenoid.

In any of the cases, the vibration source 507 structured as above is applied by power from the battery 504 and driven by the driving circuit 505, thereby causing vibration. The power consumed by the oscillation source 507 is comparatively great. Accordingly, in this embodiment the battery 504 was provided separately from the backup battery 52 (Figure 4). Due to this, when the battery 504 is consumed, a battery lid removably fitted to the back lid 502 can be opened to allow the battery 504 to be exchanged with a new one. Note that the two batteries 52 and 504 may be a same one for common use.

Also, a power line may be included in a controller cable (not shown) so that power is supplied through the power line from the image processing apparatus main body or video game machine 10 via the terminals 506 to the vibration source 507. In such a case, it is needles to say that the capacity of the power line be appropriately determined in consideration of power required for the vibration source 507.

Further, in this embodiment, the vibration source 507 was mounted on the back lid 502 so that the vibration caused by the vibration source 507 is readily delivered to a player's hand, without attenuation. That is, the vibration caused by the vibration source 507 is transmitted through the back lid 502 to the opening portion of the controller 40 in contact with the back lid 502, thus vibrating the controller 40 itself. As a result, the vibration caused by the vibration source 507 is transmitted to the player's hand gripping the controller 40. It is therefore possible to provide the vibration source 507 at an arbitrary position within the case 501, provided that the vibration by the vibration source 507 is transmitted through the controller 40 to the player's hand.

Incidentally, this embodiment provided the vibration source 507 in the controller by mounting the vibration cartridge 50 on the controller 40. However, the vibration generating circuit 53 (Figure 4), i.e. the vibration source 507, driving circuit 505, battery 504, etc. may be built in the housing of the controller 40 without using a vibration cartridge 50.

Now, the driving circuit 505 constituting the vibration generating circuit 53 will be explained in detail with reference to Figure 6. The driving circuit 505 includes a decoder comprising a NAND gate 510. This NAND gate 510 receives address data A2 - A14 through an address bus, i.e. the terminals 506 (Figure 5) from the CPU 11 (Figure 2) of the video game machine 10. In the game machine system of this embodiment, when all the addresses A0 - A15 are "1", that is, when the CPU 11 designates an address range

FFFF, a vibration mode is established to output data from the CPU 11 to drive a vibration source 507. That is, if the CPU address FFFF is designated, the output of the decoder, i.e. NAND gate 510, becomes "0". This output of the NAND gate 510 is supplied to a NAND gate 511. Since the NAND gate 511 is further supplied with a write signal-WE and a chip enable signal CE from the CPU 11, the NAND gate 511 responds to the output of the NAND gate 510 and the signals-WE and CE, to supply a latch signal to a latch 512. Consequently, when the CPU 11 designates the FFFF address, i.e. in the vibration mode, the latch 512 latches CPU data D0 through the data bus or terminals 506. This CPU data D0 is outputted as "1" when vibration is to be caused by the vibration source 507, and "0" when no vibration is to be generated. The latch 512 has an output connected to a base of a drive transistor 514 through a resistor 513. When the output of the latch 512 is "1", the transistor 514 is turned on, whereas when the output is "0" the transistor is turned off. The turning-on of the transistor 514 causes a drive current to flow from the battery 504 to the vibration source 507 (vibration motor). Thus, vibration is generated by the drive source 507.

Figure 7 is a memory map showing a memory space of the external ROM 21 incorporated in the ROM cartridge 20 (Figure 1). The external ROM 21, for example, includes a plurality of memory areas (hereinafter referred to merely as "area") such as a program area 22, a character code area 23, an image data area 24 and a sound memory area 25, as shown in Figure 5, thereby previously storing various program in a fixed manner.

The program area 22 is stored with programs required to process for game images, and game data and the like in accordance with a game content. Specifically, the program area 22 includes memory areas 12a - 22p to previously store operating programs for the CPU 11 in a fixed manner. A main program area 22a is stored with a main routine

processing program, for example, for a game shown in Figure 9 stated later. A control pad data determining program area 22d is stored with a program for processing operational data on the controller 40. A write program area 22c is stored with a write program by which the CPU 11 causes the RCP 12 to perform writing into a frame memory and Z buffer. For example, the write program area 22c is stored with a program to write, into an image data area 201 (Figure 8) of the RAM 14, chrominance data as image data based on texture data for a plurality of movable objects or background objects to be displayed in one background scene. A move program area 22d is stored with a control program by which the CPU 11 causes the RCP 12 to vary the position of a moving body in a three-dimensional space. A camera control program area 22e is stored with a camera control program that controls as to in which direction and/or position the movable objects including player object or the background objects are to be photographed in the three-dimensional space. A player object program area 22f is stored with a program that controls display of an object operated by the player. An enemy object program area 22g is stored with a program that controls display of an enemy object to make attacking on the player object. A background program area 22h is stored with a background creating program by which the CPU causes the RCP 12 to create a three-dimensional background scene.

The character code area 23 is an area to store a plurality of kinds of character codes, e.g. a plurality of kinds of character dot data corresponding to codes. The character code data stored in the character code area 23 is utilized to display an instructing text to the player in the process of a game.

An image data area 24 is stored with image data, such as coordinate data of a plurality of polygons for each of the background object and/or movable objects, and texture data, and also a display control program to display these objects stationary at a

predetermined position or in a moving state.

A sound memory area 25 is stored with sound data, such as phrases for outputting in sound the above message, effect sounds, game musics, etc., in a manner appropriate for a scene.

Incidentally, the memory medium or external memory device may use various kinds of memory mediums, such as CD-ROMs or magnetic discs, in place of or in addition to the ROM cartridge 20. In such a case, a disc drive 29 (Figure 2) is provided in order to read or write, if required, various data (including program data and data for image presentation) for a game from or onto an optical or magnetical disc memory medium such as a CD-ROM or magnetic disc. The disc drive 29 reads data out of a magnetic disc or optical disc magnetically or optically memorizing program data similarly to the external ROM 21, and transfer the same data to the RAM 14.

Figure 8 is a memory map showing a memory space of the RAM 14. The RAM 14 includes an image data area 201 and a program area 202. The image data area 201 includes, though not shown, a frame memory area for temporarily storing 1 frame of image data and a Z buffer area for storing the depth data on a dot basis in the frame memory area. The program data area 202 is an area for temporarily storing a program. The program data, allocated to the area (Figure 7) of the ROM 21, is temporarily memorized, as required, on the program data area 202 so that the CPU 11 and the RCP 12 (Figure 2) can proceed with a game by accessing to the program area of the RAM 14. Similarly, the image data area 201 is an area for temporarily memorizing, as required, the image data stored in the ROM 21, which can be directly accessed by the CPU 11 or the RCP 12. That is, the image data area 201 memorizes coordinate data and texture data of a plurality of polygons constituting stationary objects and/or movable objects stored, for game image display, in the external ROM 21. Prior to image processing, one course or

stages of data, for example, is transferred from the external ROM 21 to the image data area 201.

A controller data memorizing area 141 temporarily memorizes operating state data representative of an operating state read from the controller 40.

Also, a flag/register area 142 sets a flag as required or memorizes a variable or constant while the CPU 11 is executing a program. The flags that can be set in this flag/register area 143 includes a vibration game flag F1, a hit flag F2, a preceding-frame flag F3 and a vibration flag F4.

The vibration game flag F1 is to represent as to whether a game now being played includes a scene that vibration is to be generated by the vibration source 507 of the vibration cartridge 50. The vibration game flag F1 is set at "1" when such a scene exists, and "0" for a case other than that case. The hit flag F2 is set at "1" when two objects becomes collision or contact due to a hit determining routine (Figure 11) stated later, and "0" for a case other than that case. The preceding-frame F3 is to set whether the hit flag F2 is set to "1" at a frame previous by 1 frame on the display 30, that is, whether two objects came into collision or contact with each other at an immediately preceding frame. The preceding frame F3 is set at "1" when a hit determination is made at the immediately preceding frame, and "0" for a case other than that case. The vibration flag F4 is set at "1" when vibration is to be generated by the vibration source 507, and "0" for a case other than that case.

A variable n_add is an increment value for each frame, to increase a count value n of a counter CT in a vibration generating condition detecting routine (Figure 10) stated after, while a variable n_add_m is a value varied by the increment value n_add. The counter CT has a data size of 32 bits, so that it drives vibration source 507 (Figure 5) to generate vibration when its count value n exceeds "255" as in a 256 (8-bit binary) counter.

Variables ax, ay and az are respectively acceleration components in directions of an X-axis, Y-axis and Z-axis of the player object. Variables fx, fy and fz are values respectively multiplied of the acceleration components ax, ay and az by constants accx, accy and accz.

Incidentally, the variables n_add and n_add_m may be a constant. In the embodiment, the variable n_add is set at "255" or "150". Meanwhile, the variable n_add m is set, for example, at "20" or "10".

Figure 9 is a main flowchart for the video game system of this embodiment. If a power is turned on, the CPU, at a first step S1, sets the video game machine 10 into a predetermined initial state. For example, the CPU 11 transfers a start program, among the game programs stored in the program area 22 of the external ROM 21, to the program area 202 of the RAM 14, and sets each parameter at an initial value, thereafter sequentially executing steps of Figure 9.

The operation of the main flowchart of Figure 9 is executed, for example, every 1 frame (1/60 second) or every two or three frames, wherein steps S2 - S14 are repeatedly executed before the course is cleared. If the game becomes over without success of course clear, a game-over process is effected at a step S15 following the step S14. If course clear is successfully done, the process returns from the step S13 to the step S1.

That is, at the step S1, display is made for a game course screen and/or course choosing screen. However, where a game is started after turning on a power, a first course screen is displayed. If the first course is cleared, a next course is set.

Following the step S1, a controller process is effected at a step S2. In this process, it is detected whether any of the joystick 45, the cross switch 46 and the switches 47A - 47Z of the controller 40 is operated or not. Detected data (controller data) on this operating state is read in, and the controller data thus read is written into the controller

data area 141 of the RAM 14.

At a step S3, a process for displaying the player object is performed. This process is basically to vary the position, direction, shape and location of the player object depending upon an operating state of the joystick 45 manipulated by the player and the presence or absence of attacks by an enemy. For example, polygon data to be varied is determined by calculation based on a program transferred from the memory area 22f (Figure 7) of the external ROM 21, polygon data of the player object transferred from the memory area 24, and controller data, i.e. an operating state of the joystick 45. A plurality of polygons thus obtained are given colors due to a picture data.

At a step S4, a camera process is performed. For example, calculation is made for a visual point to the respective objects such that a visual line or field as viewed through a camera finder is in an angle at which the player designates by a joystick 45.

At a step S5, an enemy object process is performed. This process is effected based on the polygon data of an enemy object transferred from the memory area 22g and the memory area 24 (Figure 2) and according to a program partly transferred. For example, an enemy object is determined in display position and/or shape by calculating the polygon data such that the enemy object moves to attack the player object or block against its advancing while judging movement of the player object, thereby displaying an enemy object image thus varied. Due to this, the enemy object will move such that it has a certain effect upon the player object.

At a step S6, a background (stationary) object process is performed. This process is to calculate a display position and shape of a stationary object based on a program partly transferred from the memory area 22h and polygon data of a stationary object transferred from the memory area 24 (Figure 2).

At a step S7, the RSP 122 performs a rendering process. That is, the RCP 12

performs a conversion process (coordinate transformation process and frame memory rendering process), under control of the CPU 11, on image data for displaying a movable object and a stationary object based on the respective texture data for the movable object, such as an enemy object, the player object, etc. and the stationary object, such as a background, memorized in the image data area 201 of the RAM 14. Specifically, colors are put to a plurality of polygons for each of the movable objects and the stationary objects.

At a step S8, the CPU 11 performs a sound process based on sound data, such as of messages, musics, effect sounds, etc.

At a next step S9, the CPU 11 performs a vibration condition detecting process. That is, the RCP 12 at this step S9 executes a subroutine of Figure 10, based on the image data of the player object and the enemy object or the stationary object or based on the operating data supplied from the controller 40, thereby detecting whether a condition under which vibration is to be generated by the vibration source 507 is established or not. Incidentally, a vibration condition detecting subroutine will be explained in detail later with reference to Figure 10.

At a next step S10, the CPU 11 responds to a detection of a vibration generating condition by the step S9, to execute a subroutine shown in Figure 12, thereby causing vibration by the vibration source 507. This step S10 will be also explained in detail with difference to Figure 12.

At a step S11, the CPU 11 reads the image data memorized in the frame memory area of the RAM 14 as a result of the rendering process by the step S7. This causes the player object, the movable object, the stationary object, and the like to be displayed on a display screen of the display 30 (Figure 1, Figure 2).

At a step S12, the sound data that has been sound processed at the step S8 by the

RCP 12 is read out to thereby output a sound, such as a music, effect sound conversation or the like.

It is determined at a step S13 whether the course is cleared or not (course clear detection). If the course is not cleared, it is then determined at a step S14 whether it is game-over or not. If it is not game-over, the process returns to the step S2, to repeat the steps S2 - F14 until a condition of game-over is detected. If detecting a game-over condition that a number of mistakes permitted for a player reaches a predetermined number of times or the life of the player object is consumed by a predetermined amount, then a game-over process is carried out at a succeeding step S15 to choose game continuation, backup data saving , or the like.

Incidentally, if the condition to clear the course is detected (e.g. defeating a boss) at the step S13, the process returns to the step S1 after course-clear processing.

Referring to Figure 10, at a first step S101 of a vibration generating condition detecting subroutine, the CPU 11 determines whether or not the player object hits (collides or contacts) against an influencing object or obstacle (other movable objects, stationary objects such as ground, sea surface, wall, enemy objects, attacking objects, etc.). This hit detection is executed according to a subroutine of Figure 11.

At a step S201 in Figure 11, the CPU 11 determines whether ABS (OBJ2x − OBJ1x) ≦ OBJ1r is satisfied or not. That is, whether two objects are hit against each other on an X-coordinate system or not. OBJ1 is an object to be hit-determined, which in this embodiment is the player object. OBJ2 is an object moving toward OBJ1, which in this embodiment a fellow object, an enemy object, a stationary object and an attacking object launched by the enemy object. OBJ1x is an X-coordinate value of OBJ1, while OBJ2x is an X-coordinate value of OBJ2. OBJ1x and OBJ2x may be on a game space coordinate or player coordinate, provided that they are an X-coordinate value of a same

coordinate system. ABS ( ) represents an absolute value of a numeral within ( ). OBJ1r is a value representative of a half length of a side of a cube when OBJ1 is considered as a cube. In other words, OBJ1r is a value indicating a hit range for OBJ1. If ABS (OBJ2x - OBJ1x) ≦ OBJ1r is satisfied, the process proceeds to a step S202.

At a step S202, the CPU 11 determines whether ABS(OBJ2y - OBJ1Y) ≦ OBJ1r is satisfied or not, that is, whether there is hit between two objects on a Y-coordinate system or not. OBJ1y is a Y-coordinate value of OBJ1, while OBJ2y is a Y-coordinate value of OBJ2. OBJ1y and OBJ2y may be on a game space coordinate or player coordinate, provided that they are in a Y-coordinate value on a same coordinate system. If ABS (OBJ2y - OBJ1y) ≦ OBJ1r is fulfilled, the process advances to a step S203.

At the step S203, the CPU 11 determines whether ABS (OBJ2z - OBJ1z) ≦ OBJ1r stands or not, that is, whether there is a hit between two objects on the Z coordinate system or not. OBJ1z is a Z-coordinate value of OBJ1, while OBJ2z is a Z-coordinate value of OBJ2. OBJ1z and OBJ2z may be on a game space coordinate or player coordinate, provided that they are in a Z-coordinate value on a same coordinate system. If ABS (OBJ2z - OBJ1z) ≦ OBJ1r is satisfied, the process proceeds to a step S204.

At the step S204, the CPU 11 determines that there is a hit between OBJ2 and OBJ1, and sets a hit flag F2 in the flag area 142 of the RAM 14 to "1".

On the other hand, if ABS (OBJ2x - OBJ1x) ≦ OBJ1r is not satisfied at the step S201, the process returns to a former routine. If ABS (OBJ2y - OBJ1y) ≦ OBJ1r is not profiled at the step S202, the process returns to the former routine. If ABS (OBJ2z - OBJ1z) ≦ OBJ1r is not satisfied at the step S203, the process returns to the former routine.

At the step S101 in Figure 10, if it is detected that the player object is not hit by another object, that is, if "NO" is determined at the step S101, the CPU 11 at a next step

S102 determines whether the player manipulates the controller 40 to start acceleration of the player object or not. For example, the acceleration start is effected by depressing an A button 47a (Figure 1), where the player object is a "jet ski" in a "wave race". Also, if the player object is "Mario", the acceleration start is by inclining the joystick 45 (Figure 1) frontward. Accordingly, the CPU 11 at this step S102 makes reference to the data in the controlller data area 141 of the RAM 14, and determines whether there is an operation of the A button 47A or joystick 45 or not.

At the step S102, if "NO" is determine, the CPU 11 at a next step S103 determines whether the player object ("jet ski" in the "wave race" in this embodiment) is in contact with a water surface or not. At the step S103, the hit detecting subroutine of Figure 11 is utilized in order to determine whether the player object ("jet ski") is in contact with the water surface or not.

In the vibration generating condition detecting subroutine shown in Figure 10, detection is made for any of the three vibration generating conditions, i.e. any of the steps S101, S102, and S103, as explained above. If "NO" is determined at any of the steps S101, S102 and S103, that is, if no vibration generating condition is detected, the CPU 11 resets, at a step S104 in Figure 10, the register value n_add of the flag/register area 142 of the RAM 14 to "0". At same time, the CPU 11 at next step S105 clear off the preceding frame flag F3 of the flag/register area 142 to "0". That is, the count value n of the counter CT is added by "0" for each frame, in order to set the increment value n_add of the counter CT at "0". In other words, if "NO" is determined at all the steps S101, S102 and S103, the counter CT is not incremented at all. As will be stated later, if the count value of the counter CT exceeds, for example, "255", vibration will occur. Accordingly, in the above case no vibration is generated by the vibration source 507.

Incidentally, since "NO" is determined at the above-mentioned step S101, the

preceding-frame flag F3 is reset at the step S105.

If "YES" is determined at any of the steps S101, S102 and S103, a vibration generating condition is established and a vibration generating process is effected according to the corresponding vibration generating condition.

That is, if it is determined, at the step S101, that the player object is hit against another object, steps S106 to S110 are executed to generate intense vibration from the vibration source 507. Meanwhile, an acceleration start is detected at the step S102, weak vibration is generated by steps S111 to S113. If the player object (e.g. "jet ski") is in contact with a water surface is detected at the step S103, steps S114 to S117 are executed to generate weak vibration representing a state that the "jet ski" moves bounding on waves.

If "YES" is determined at the above-mentioned step S101, that is, if the player object comes into collision or contact with another object is determined, the CPU 11 determines whether the preceding-frame flag F3 is at "0" or not. That is, at this step S106 it is determined whether the player object collided against another object also at the preceding frame or not. If at the step S106 "YES" is determined, that is, if it is determined there was no collision or contact of the player object at the preceding frame but there is detection at a current frame of collision or contact of the player object with another object, the CPU 11 at a next step S107 sets the increment value n_add of the flag/register area 142 of the RAM 14, for example, to "255" in order to generate intense vibration.

That is, where the player object comes into collision or contact with an enemy object as shown in Figure 13, or where the player object crashes into or contacts an enemy object or stationary object as shown in Figure 14, a greater increment value n_add is set at a step S107 in order to cause intense vibration.

Then the CPU 11 at a step S108 sets a variation value n_add_m of the increment

value n_add at "20". When there is collision or contact with another object, the increment value n_add is set to "255" at the step S107 and the variation value n_add_m is set to "20" at the step S108 in order to generate intense vibration. If the variation value n_add_m is great in value, the increment value n_add becomes "0" in a brief time, while if the variation value n_add_m is small, it takes a long time for the increment value n_add to reach "0". Therefore, when the variation value n_add_m is great, the vibration due to the vibration source 107 continues for a long time. At a step S109 following the step S108, the CPU 11 sets the preceding-frame flag F3 to "1". That is, since at the current frame the player object is detected of hit, the previous frame flag F3 is set to "1" at this time point.

Incidentally, if "NO" is detected at the above-mentioned step S106, that is, if the preceding-frame flag F3 is "1", that is, if there is also a detection of collision or contact of the player object at the preceding frame, the increment value n_add is reset to "0" at a step S110. That is, when the player object comes into contact or collision with another object also at the preceding frame, the increment value n_add is rendered "0" at the step S110 in order to prevent vibration from continuously occurring.

At the step S102, if an acceleration start of the player object is detected, the CPU 11 sets the increment value n_add, for example, to "150" at a next step S111 to generate weak vibration. That is, when an acceleration of the player object is started, the increment value n_add of the counter CT is set to "150" smaller than the value of the step S107 in order to generate comparatively small vibration differently from the case of the above-stated hit detection. The CPU 11 then sets the variation value n_add_m to "10" at a step S112. At a step S113, the preceding-frame flag F3 is rendered "0". That is, this step S113 is a step to be executed when "NO" is determined at the above-mentioned step S101, and accordingly the preceding-frame flag F3 is rendered "0" in a manner different from that of the above step S109.

Further, if it is detected at the step S103 that the player object, i.e. "jet ski" in this embodiment, is in contact with a water surface, calculations are made for multiplying values fx, fy and fz at a next step S114. The CPU 11 at a step S115 calculates a square root (decimal fraction truncated) of "fx × fx + fy × fy + fz × fz" as an increment value n_add of the counter CT. That is, at these steps S114 and S115, acceleration components ax, ay and az of the player object in the respective X-axis direction, Y-axis direction and Z-axis direction are determined to determine values fx, fy and fz respectively proportional to the acceleration components. An increment value n_add in a range of "0" - "255" is calculated, depending upon these proportional values and multiplying values. At a step S116, the variation value n_add_m of the increment value n_add is set to, for example, "10". That is, where the player object "jet ski" is in contact with the water surface, the variation value n_add_m is set at a relatively small value "10" in order to give impact for a comparatively long time. In also this case, since "NO" is determined at the above step S101, the preceding-frame flag F3 is rendered "0" at a succeeding step S117.

At a first step S120 in Figure 12 showing a vibration generating subroutine, the CPU 11 determines whether the vibration game flag F1 in the flag/register area 142 of the RAM 14 is at "1" or not, that is, whether a game now being played is a game involving vibration or not. If "YES" is determined at this step S120, the CPU 11 makes reference to the controller data area 141 of the RAM 14, and determines whether a vibration cartridge 50 (Figure 1, Figure 4) is loaded on the controller 40 or not. If this game is a game with vibration and a vibration cartridge 50 is loaded on the controller 40, the CPU 11 determines at a next step S122 whether or not vibration has to be forcibly stopped despite a vibration generating condition is established, that is, whether vibration is reset or not.

If "NO" is determined at this step S122, the CPU 11 renders the count value n of the counter CT as "n + n_add" at a next step S123. That is, the count value n of the

counter CT is increased according to an increment value n_add.

At a next step S124, the increment value n_add is modified according to a variation value n_add_m. That is, the increment value n_add is subtracted by the variation value n_add_m, at this step S124, at every 1 frame of the display 30. Accordingly, the count value n of the counter CT has an increment value decreasing with progress of frames, and the count value n finally does not increase. In other words, the vibration due to the vibration source 507 is initially large (intense) and gradually decreases (weakens) to ultimate no vibration.

At a step S125, it is determined whether the count value n of the counter CT exceeds "255" or not. if "YES" is determined at this step S125, the count value of the counter CT is rendered "n - 256". That is, if it is determined at the step S125 that the count value n of the counter CT exceeds "255", the count value n is subtracted by "256" at a next step S126. At a step S127, the CPU 11 sets the vibration flag F4 at "1". Since the count value n of the counter CT exceeds "255" is detected at the above step S125, the vibration flag F4 at this step S127 is set at "1".

At a next step S128, it is determined whether the vibration flag F4 is at "1" or not. Since vibration is generated by setting the vibration flag F4 at "1" at the above step S127, the CPU 11 at a next step S129 outputs "1" to all the addresses A2 - A14 except for an address A15, and outputs a write signal and a chip enable signal. Accordingly, at this step S129 a recorder or NAND gate 511 (Figure 6) has an output signal, and the latch 512 latches data bit D0 of the CPU 11. Since at the step S129 vibration has to be generated by the vibration source 507, the data bit D0 of the CPU 11 is outputted as "1". Accordingly, "1" is latched by the latch 512 (Figure 6). In response thereto, the transistor 514 is turned on to supply power from the battery 504 to the vibration source or vibrating motor 507, thereby causing vibration in the vibration source 107 or vibration cartridge 50, i.e. on the

controller 40.

Incidentally, if "NO" is determined at the step S125, the vibration flag F4 is reset at "0" at a step S130. That is, when the count value n of the counter CT does not exceed "255", the vibration flag F4 is kept in a reset state.

At the step S127, when the vibration flag F4 is not at "1", that is, when the vibration flag F4 is at "0", the CPU 11 outputs "0" to the data bit "D0" at a step S131, in order to stop the vibration. Consequently, "0" is latched by the latch 512, and the transistor 514 is turned off. Accordingly, the vibration source 507 has no current, and no vibration is generated by the vibration source 507.

For example, if the player object comes into collision or contact with an enemy object or stationary object (Figure 13, Figure 14), "255" is set as an increment value n add of the count value n of the counter CT at the step S107, and "20" is set as a variation value n_add_m at the step S108. The count value n of the counter CT exceeds "255" over consecutive 4 frames excepting the first frame as shown in Table 1 and Figure 15. Therefore, the vibration source 507 is continuously driven over the consecutive 4 frames, and thereafter the count value n exceeds "255" every two frames. The vibration source 507 generates a comparatively intense vibration driven every two frames.

Table 1

	INTENSE VIBRATION EXAMPLE			
FRAME	**n**	**n_add**	**n_add_m**	**VIBRATION SIGNAL**
1	255	255	20	0
2	234	235	20	1
3	193	215	20	1
4	132	195	20	1
5	51	175	20	1
6	206	155	20	0
7	85	135	20	1
8	200	115	20	0
9	39	95	20	1
10	114	75	20	0
11	169	55	20	0
12	204	35	20	0
13	219	15	20	0
14	219	0	20	0
15	219	0	20	0
16	219	0	20	0
17	219	0	20	0
18	219	0	20	0
19	219	0	20	0
20	219	0	20	0

However, when starting an acceleration for the player object, comparatively weak vibration is generated. Accordingly, "150" is set as an increment value n_add at a step S111, and "10" is set as a variation value n_add_m at a step S112. Consequently, in this case the count value n exceeds "255" every two frames as shown in Table 2 (Figure 16) so that the vibration source 507 is driven every two frames and the following 2 frames are suspended of vibration. At a next 1 frame the vibration source 507 is driven, and thereafter the vibration is suspended for 2 frames.

Table 2

	WEAK VIBRATION EXAMPLE			
FRAME	n	n_add	n_add_m	VIBRATION SIGNAL
1	150	150	10	0
2	34	140	10	1
3	164	130	10	0
4	28	120	10	1
5	138	110	10	0
6	238	100	10	0
7	72	90	10	1
8	152	80	10	0
9	222	70	10	0
10	26	60	10	1
11	76	50	10	0
12	116	40	10	0
13	146	30	10	0
14	166	20	10	0
15	176	10	10	0
16	176	0	10	0
17	176	0	10	0
18	176	0	10	0
19	176	0	10	0
20	176	0	10	0

That is, where generating intense vibration, the vibration source 507 is continuously driven over several frames and then the vibration is gradually decreased as if driven every 2 frames. When weak vibration is to be generated, the driving source 507 is driven every 2 frames and then the vibration is gradually decreased in a manner driven every 3 frames. However, it is needless to say that the increment value n_add or the variation value n_add_m, that is, the vibration generating pattern as above, can be set in an arbitrary manner.

Incidentally, under a third vibration generating condition, i.e. where the player object ("jet ski") is in contact with a water surface, the increment value n_add is set as a

function of acceleration of the player object, generating intense or weak vibration depending upon the acceleration.

Incidentally, in the above embodiment the count value n of the counter CT is varied according to an increment value n_add or an variation value n_add_m, by executing the flowchart of Figure 12, in order to generate intense or weak vibration as shown in Figure 15 or Figure 16. When the count value n exceeds "255", the vibration flag F4 is set to "1" to drive the vibration source 507. That is, in the above embodiment whether to generate vibration by the vibration source 507 is determined by calculation in a real time manner.

Contrary to this, intense and weak vibration pattern data, read out of the program area 22a of the external ROM 21, may be respectively memorized on the flag/register area 142 of the RAM 14, as shown, for example, in Figure 17. By selectively reading out these vibration pattern data, the vibration source 507 may be controlled based on these pattern data. It can be considered that the intense vibration pattern data uses a vibration signal "0111101010000000" in a frame sequence of frames 1 -16 in the afore-said Table 1. Meanwhile, the weak vibration pattern data may use a vibration signal "0101001001000000" in a frame sequence of frames 1 - 16 in the afore-mentioned Table 2. Incidentally, medium vibration pattern data may be memorized as required.

For example, when the player object is detected of its collision or contact at the step S101 in Figure 10, if intense vibration pattern data is read out, the vibration source 507 is not driven by a first frame. However, the vibration source 507 is continuously driven over the following 4 frames, and then at every 2 frames, thereafter being ceased of drive. For example, when an acceleration start is detected at the step S102 in Figure 10 or a contact with a water surface is detected at the step S103, weak vibration pattern data is read out. Consequently, the vibration source 507 is driven at every 2 frames during the

first 4 frames, and then driven at every 3 frames, thereafter being ceased of drive.

In the above embodiment, when the player object is hit by another object or the like, vibration is generated by the vibration source 507 of the vibration cartridge 50 loaded on the controller 40. Vibration may be visually given to game images in timing related to this vibration. In such a case, the game-image vibration on the display 30 can be perceived with higher rapidity than the mechanical vibration caused by the vibration cartridge 50 of the controller 40. Accordingly, mechanical vibration may be generated by the vibration cartridge 50 at the step S10 in Figure 9, and thereafter vibration is visually given to game images with a delay of 1 or 2 frames or more at the step S4 or S7 in Figure 9.

For example, when it is detected at the step S101 of Figure 10 that the player object comes into contact or collision against another object, since intense vibration is generated on a game images, the display 30 is vibrated at the entire screen. In this case, the visual coordinate for a camera stated before may be varied bit by bit during the camera process of the step S4 in Figure 9.

When a condition of generating weak vibration is detected at the step S102 or S103 in Figure 10, weak vibration is generated on the game image. The player object image only is given vibration that is displayed on the display 30. In such a case, a plurality of polygon sets, constituting the player object, may be varied in a center coordinate position bit by bit in the rendering process at the step S7 in Figure 9.

Incidentally, whether to give vibration to a game image may be determined by a condition whether "F4 = 1" is detected at the step S128 in Figure 12. That is, when the vibration flag F4 is at "1", vibration is given to the game image, while when the flag is at "0", no vibration is generated in the game image.

Although the present invention has been described and illustrated in detail, it is

clearly understood that the same is by way of illustration and example only and is not to be taken by way of limitation, the spirit and scope of the present invention being limited only by the terms of the appended claims.

Throughout this specification and the claims which follow, unless the context requires otherwise, the word "comprise", and variations such as "comprises" and "comprising", will be understood to imply the inclusion of a stated integer or step or group of integers or steps but not the exclusion of any other integer or step or group of integers or steps.

The reference to any prior art in this specification is not, and should not be taken as, an acknowledgment or any form of suggestion that that prior art forms part of the common general knowledge in Australia.

THE CLAIMS DEFINING THE INVENTION ARE AS FOLLOWS:

1. A video game system, having at least a video game machine, a memory medium and a controller, wherein said video game machine includes a processing means for generating game images to be displayed on a display depending upon operation data output from said controller and according to a game program memorized in said memory medium;

said controller comprising:

a housing for being grasped by a hand of a player;

a plurality of operating switches that the player uses to generate said operation data to designate a movement or motion of a player object;

a vibration source for applying vibration to said housing; and

a drive circuit for driving said vibration source in response to a command signal supplied from said video game machine;

said memory medium comprising the following programs to be executed by said processing means:

a player object image generating program for generating an image of the player object that is varied in response to at least one operation data input from at least one of said operating switches;

another object image generating program for generating an image of at least one other object that is present around the player object;

a vibration generating condition detecting program for detecting a first vibration generating condition for generating weak vibration depending upon a display state relationship between the player object image and at least one other object image, and for detecting a second vibration generating condition for generating intense vibration depending upon said display state relationship; and

a vibration control program for generating control data to turn said vibration source on or off for each display frame of said display in response to detection of one of the first and second vibration generating conditions, according to the vibration generating condition detecting program, and causing said vibration source to generate the comparatively intense vibration or the comparatively weak vibration depending on the number of frames for which said vibration source is turned on and off;

COMS ID No: SBMI-00772060 Received by IP Australia: Time (H:m) 14:54 Date (Y-M-d) 2004-05-31

said processing means controlling said vibration source to generate the comparatively weak vibration in response to detection of the first vibration generating condition and to generate the comparatively intense vibration in response to detection of the second vibration generating condition, according to said control data.

2. A video game system according to claim 1, wherein said vibration control program calculates the number of frames to turn on and off said vibration source depending upon the display state of the player object image and the other object image.

3. A video game system according to claim 1, wherein said vibration control program includes a plurality of vibration pattern data previously determined by the display state of the player object and the other object image.

4. A video game system according to claim 1, wherein said memory medium includes an image vibrating program for visually causing at least one of the player object and the other object to vibrate.

5. A video game system according to claim 3, wherein said vibration control program causes image vibration at timing later than generation timing of the mechanical vibration.

6. A video game system, having at least a video game machine and a controller, wherein said controller comprises a housing for being grasped by a hand of a player, a plurality of operating switches that the player uses to generate operation data to designate a movement or motion of a player object, a vibration source for applying vibration to said housing, and a drive circuit for driving said vibration source in response to a command signal supplied from said video game machine,

said video game machine, comprising:

a player object image generating means for generating an image of the player object that is varied in response to at least one operation data input from at least one of said operating switches;

another object image generating means for generating an image of at least one other object that is present around the player object;

COMS ID No: SBMI-00772060 Received by IP Australia: Time (H:m) 14:54 Date (Y-M-d) 2004-05-31

a vibration generating condition detecting means for detecting a first vibration generating condition for generating weak vibration depending upon a display state relationship between the player object image and at least one other object image, and for detecting a second vibration generating condition for generating intense vibration depending upon said display state relationship; and

a vibration generating means for causing through said drive circuit said vibration source to generate vibration at timing synchronous with the generation of each display frame on a display, and for generating control data to turn said vibration source on or off for each display frame of said display in response to detection of one of the first and second vibration generating conditions by the vibration generating condition detecting means, and causing said vibration source to generate the comparatively intense vibration or the comparatively weak vibration depending on the number of frames for which said vibration source is turned on and off;

said vibration generating means including a drive means for controlling said vibration source to generate the comparatively weak vibration in response to detection of the first vibration generating condition and to generate the comparatively intense vibration in response to detection of the second vibration generating condition, according to said control data.

7. A video game system according to claim 6, wherein said vibrating generating means includes a calculating means for calculating a number of frames for which said vibration source is turned on and off.

8. A video game system substantially as hereinbefore described with reference to the drawings.

DATED this 31st day of May 2004

NINTENDO CO., LTD.
By its Patent Attorneys
DAVIES COLLISON CAVE

COMS ID No: SBMI-00772060 Received by IP Australia: Time (H:m) 14:54 Date (Y-M-d) 2004-05-31

FIG. 1

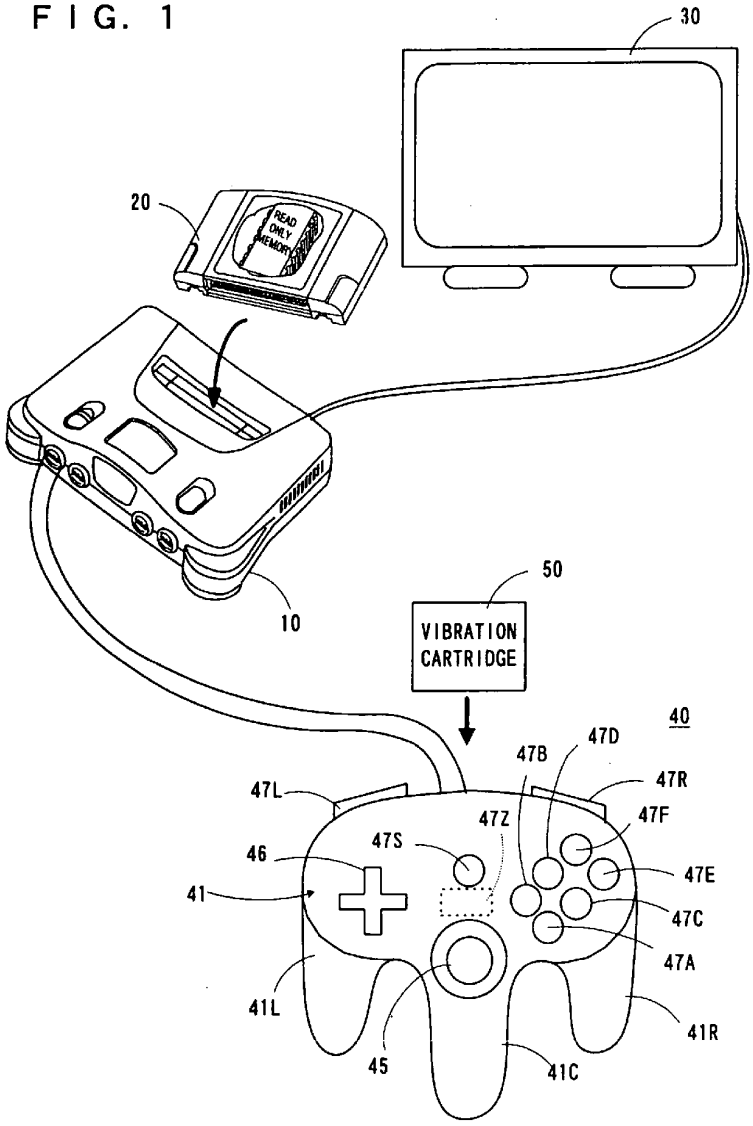

30
20
READ ONLY MEMORY
10
50
VIBRATION CARTRIDGE
40
47L
47B
47D
47R
47F
47Z
47S
46
41
47E
47C
47A
41L
41R
45
41C

FIG. 2

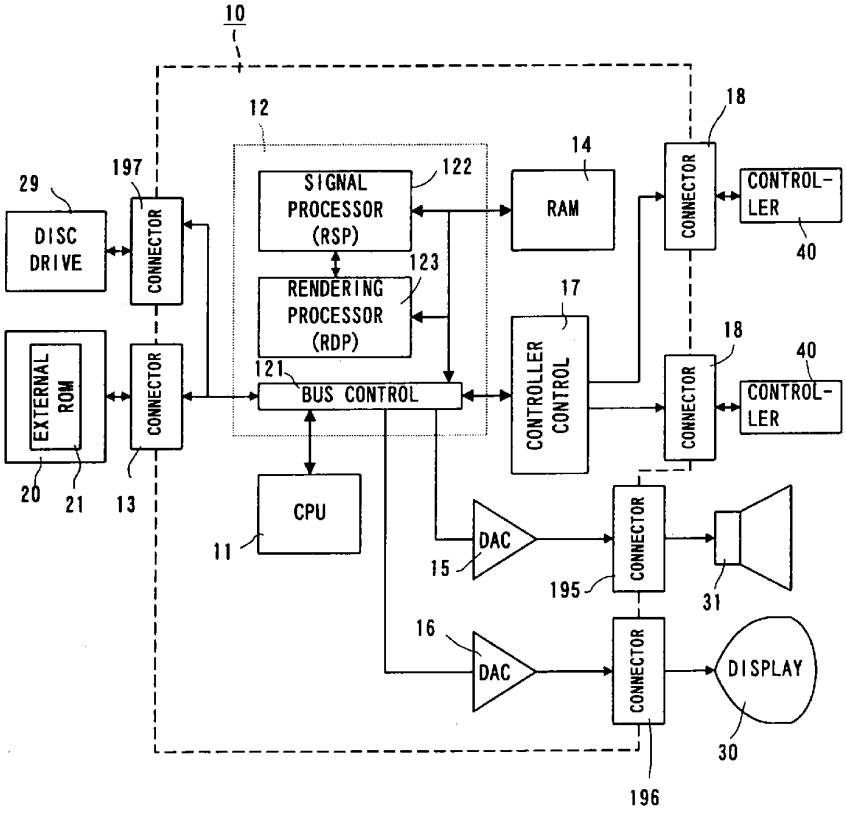

10
12
18
14
197
122
29
SIGNAL PROCESSOR (RSP)
RAM
CONNECTOR
CONTROL-LER
DISC DRIVE
CONNECTOR
123
RENDERING PROCESSOR (RDP)
17
18
40
121
BUS CONTROL
CONTROLLER CONTROL
CONNECTOR
CONTROL-LER
EXTERNAL ROM
CONNECTOR
20
21
13
CPU
11
DAC
15
CONNECTOR
195
31
16
DAC
CONNECTOR
DISPLAY
30
196

FIG. 3

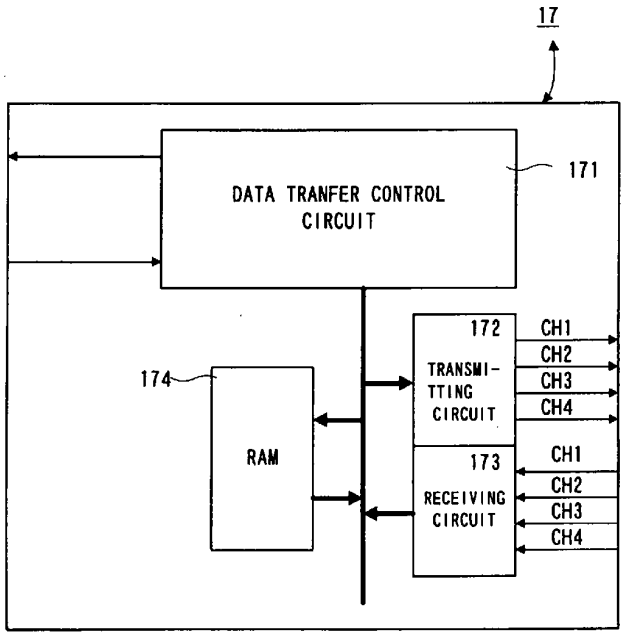

17
171
DATA TRANFER CONTROL CIRCUIT
172
TRANSMI-TTING CIRCUIT
CH1
CH2
CH3
CH4
174
RAM
173
RECEIVING CIRCUIT
CH1
CH2
CH3
CH4

FIG. 4

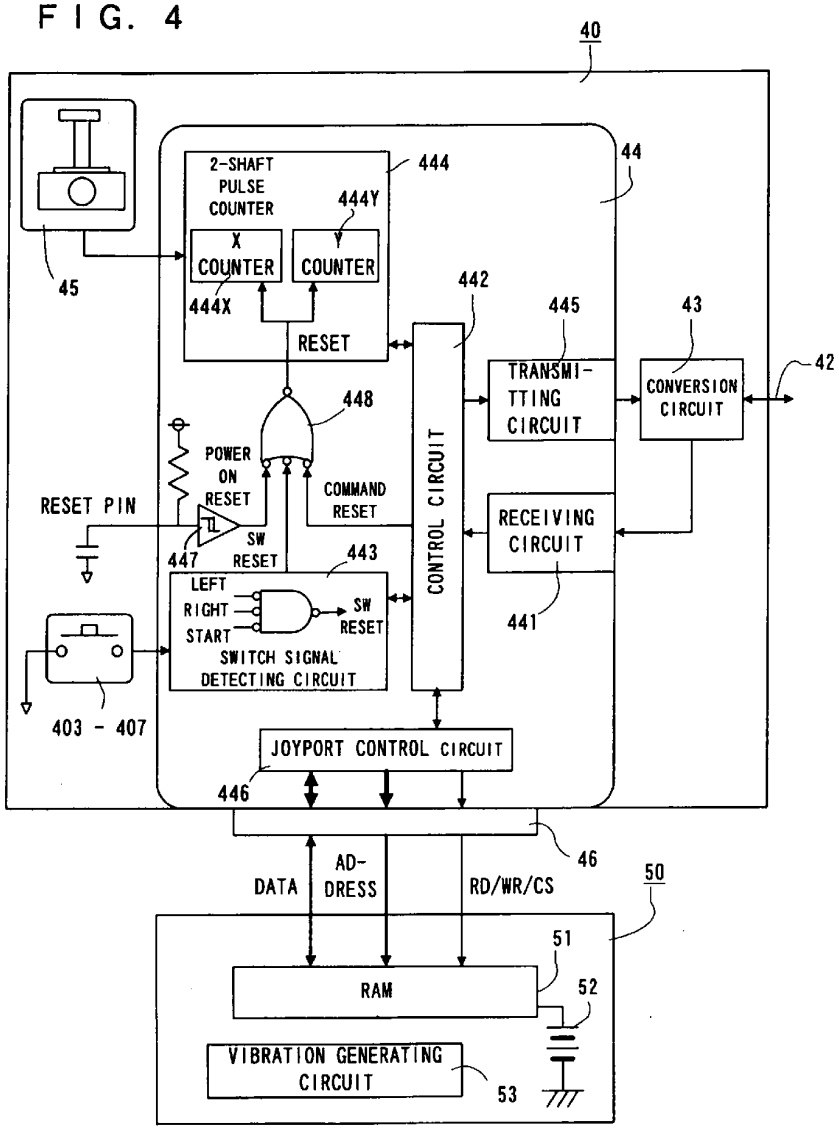

40
44
2-SHAFT PULSE COUNTER
444
444Y
X COUNTER
Y COUNTER
444X
45
RESET
442
445
43
42
TRANSMI-TTING CIRCUIT
CONVERSION CIRCUIT
448
POWER ON RESET
COMMAND RESET
RESET PIN
447
SW RESET
CONTROL CIRCUIT
RECEIVING CIRCUIT
443
LEFT
RIGHT
START
SW RESET
SWITCH SIGNAL DETECTING CIRCUIT
441
403 - 407
JOYPORT CONTROL CIRCUIT
446
46
DATA
AD-DRESS
RD/WR/CS
50
51
RAM
52
VIBRATION GENERATING CIRCUIT
53

FIG. 5

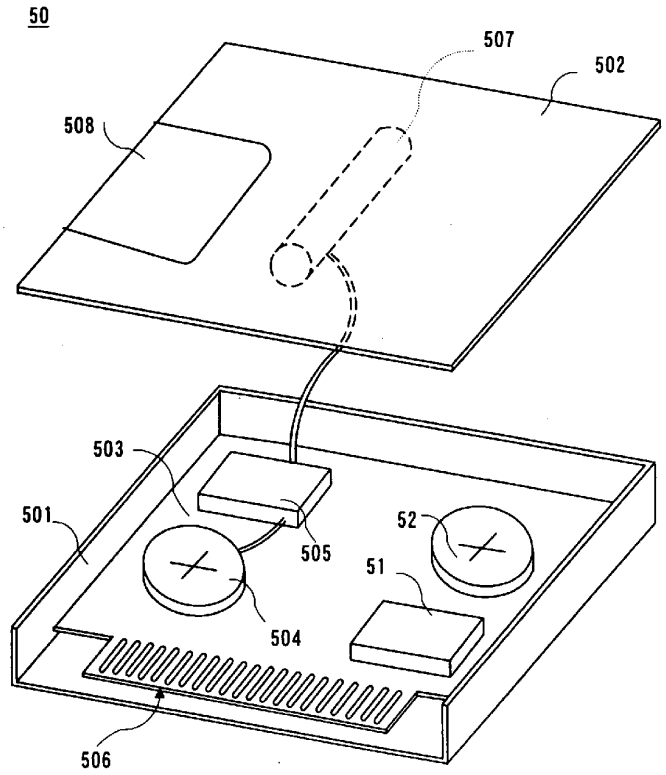


FIG. 6

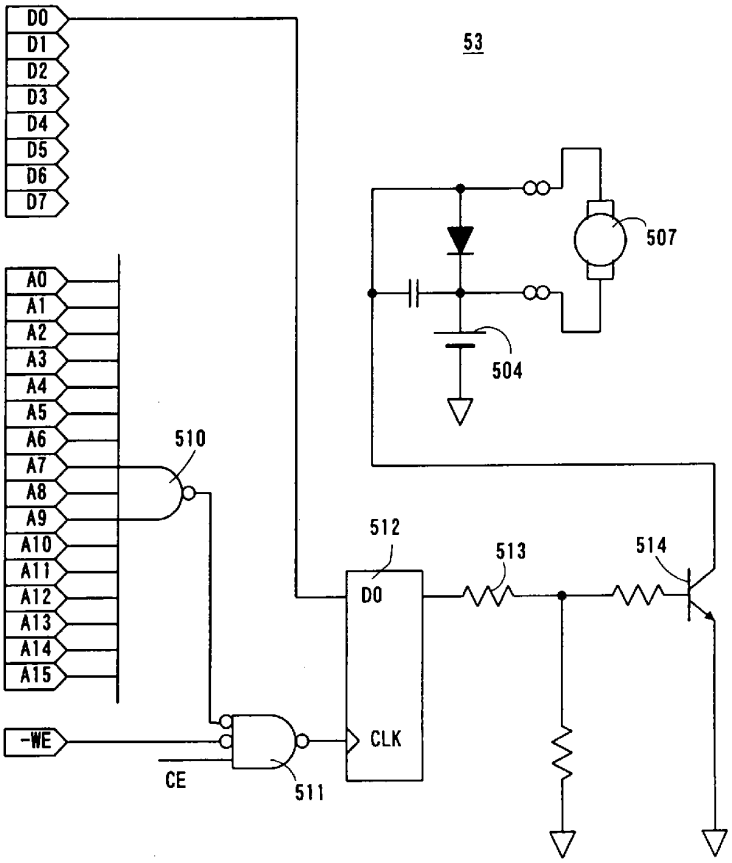

D0
D1
D2
D3
D4
D5
D6
D7
A0
A1
A2
A3
A4
A5
A6
A7
A8
A9
A10
A11
A12
A13
A14
A15
-WE
CE
510
511
512
513
514
53
507
504
DO
CLK

FIG. 7

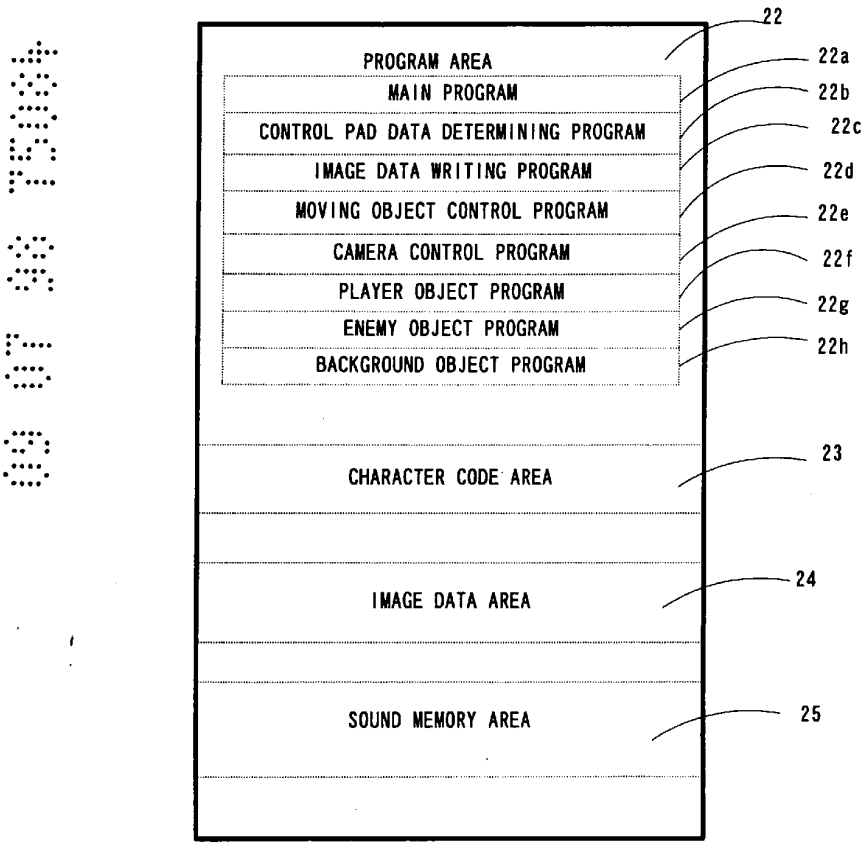

EXTERNAL ROM 21
22
PROGRAM AREA
22a
MAIN PROGRAM
22b
CONTROL PAD DATA DETERMINING PROGRAM
22c
IMAGE DATA WRITING PROGRAM
22d
MOVING OBJECT CONTROL PROGRAM
22e
CAMERA CONTROL PROGRAM
22f
PLAYER OBJECT PROGRAM
22g
ENEMY OBJECT PROGRAM
22h
BACKGROUND OBJECT PROGRAM
CHARACTER CODE AREA
23
IMAGE DATA AREA
24
SOUND MEMORY AREA
25

FIG. 8

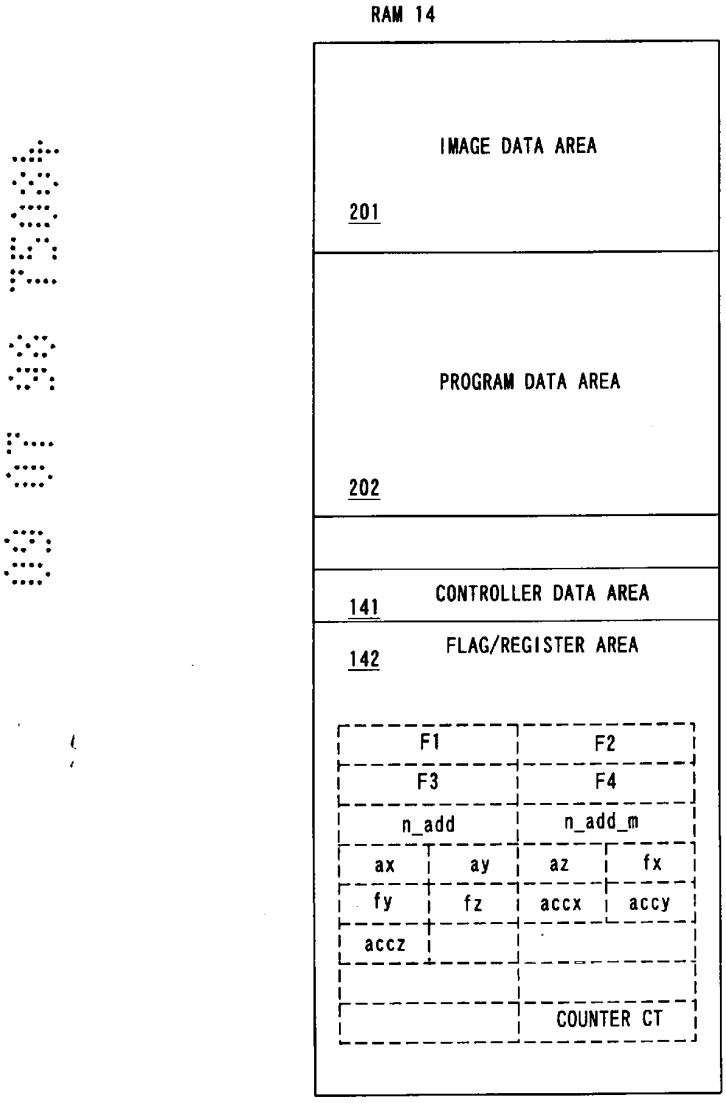

RAM 14
IMAGE DATA AREA
201
PROGRAM DATA AREA
202
CONTROLLER DATA AREA
141
FLAG/REGISTER AREA
142
F1
F2
F3
F4
n_add
n_add_m
ax
ay
az
fx
fy
fz
accx
accy
accz
COUNTER CT

FIG. 9

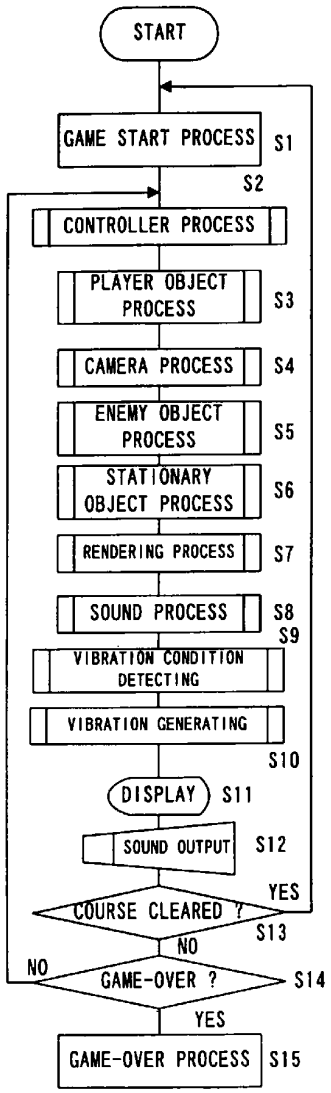

START
GAME START PROCESS
S1
S2
CONTROLLER PROCESS
PLAYER OBJECT PROCESS
S3
CAMERA PROCESS
S4
ENEMY OBJECT PROCESS
S5
STATIONARY OBJECT PROCESS
S6
RENDERING PROCESS
S7
SOUND PROCESS
S8
S9
VIBRATION CONDITION DETECTING
VIBRATION GENERATING
S10
DISPLAY
S11
SOUND OUTPUT
S12
YES
COURSE CLEARED ?
S13
NO
NO
GAME-OVER ?
S14
YES
GAME-OVER PROCESS
S15

FIG. 10

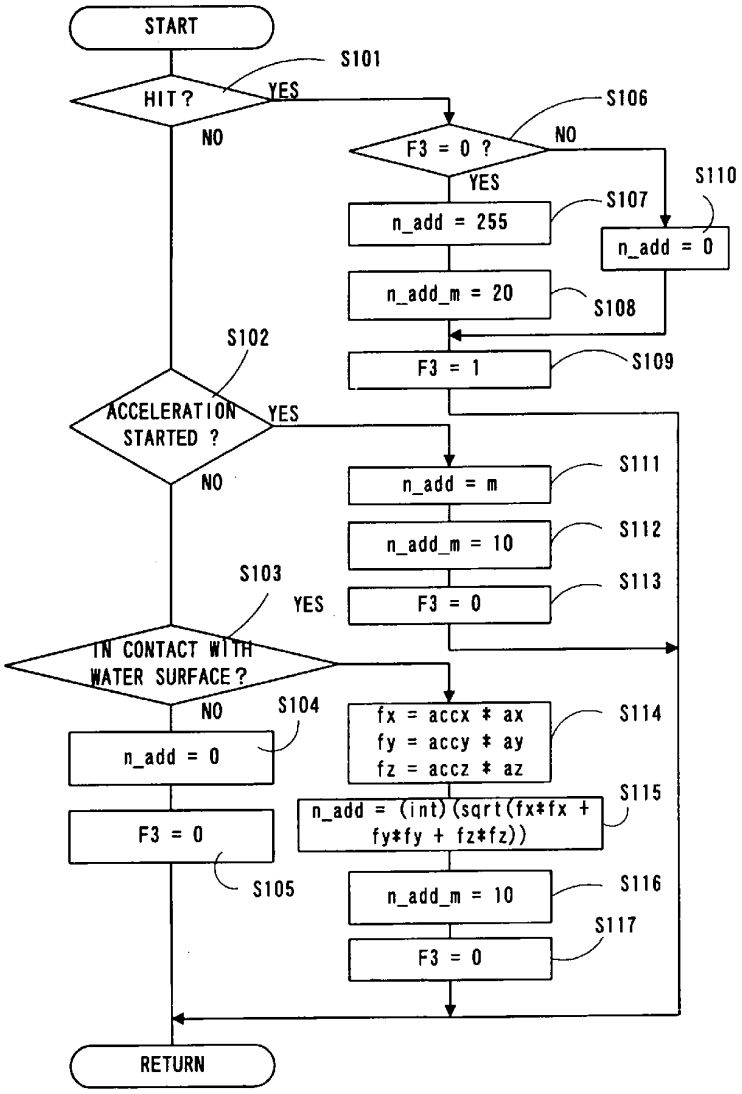

START
S101
HIT ?
YES
NO
S106
F3 = 0 ?
NO
YES
S110
S107
n_add = 255
n_add = 0
n_add_m = 20
S108
S102
F3 = 1
S109
ACCELERATION STARTED ?
YES
NO
S111
n_add = m
S112
n_add_m = 10
S103
S113
YES
F3 = 0
IN CONTACT WITH WATER SURFACE ?
NO
S104
S114
fx = accx * ax
fy = accy * ay
fz = accz * az
n_add = 0
S115
n_add = (int)(sqrt(fx*fx + fy*fy + fz*fz))
F3 = 0
S105
S116
n_add_m = 10
S117
F3 = 0
RETURN

FIG. 11

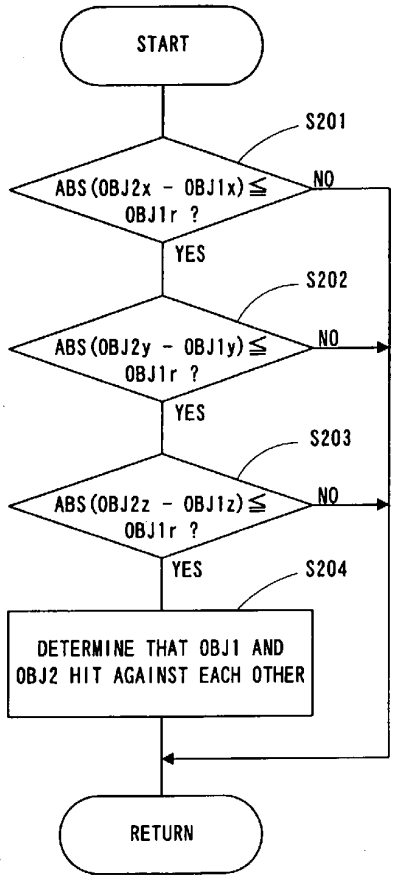

START
S201
ABS(OBJ2x - OBJ1x)≦ OBJ1r ?
NO
YES
S202
ABS(OBJ2y - OBJ1y)≦ OBJ1r ?
NO
YES
S203
ABS(OBJ2z - OBJ1z)≦ OBJ1r ?
NO
YES
S204
DETERMINE THAT OBJ1 AND OBJ2 HIT AGAINST EACH OTHER
RETURN

FIG. 12

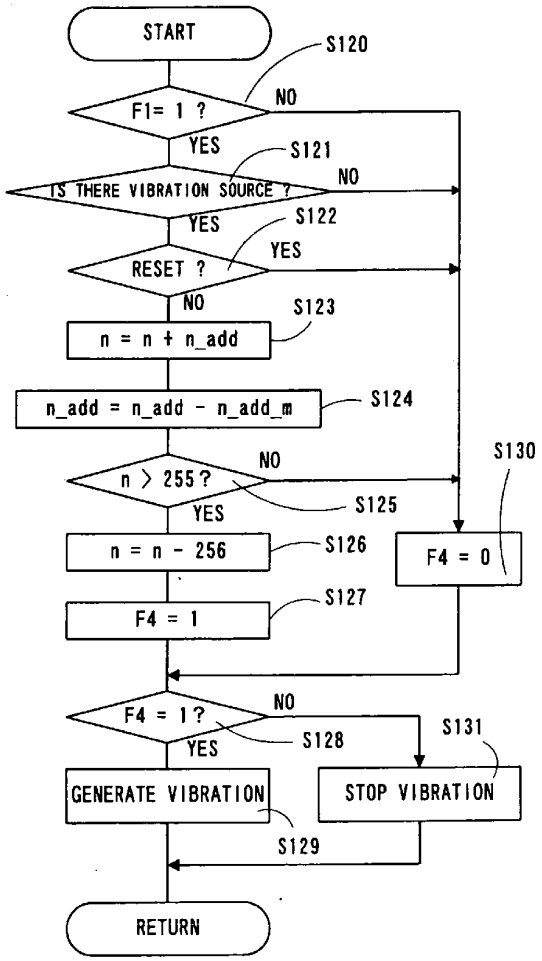

START
S120
F1= 1 ?
NO
YES
S121
IS THERE VIBRATION SOURCE ?
NO
YES
S122
RESET ?
YES
NO
S123
n = n + n_add
n_add = n_add - n_add_m
S124
n > 255 ?
NO
S130
YES
S125
n = n - 256
S126
F4 = 0
F4 = 1
S127
F4 = 1 ?
NO
S131
YES
S128
GENERATE VIBRATION
STOP VIBRATION
S129
RETURN

FIG. 13

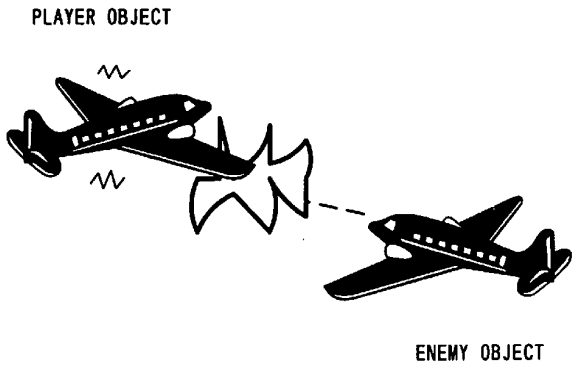


FIG. 14

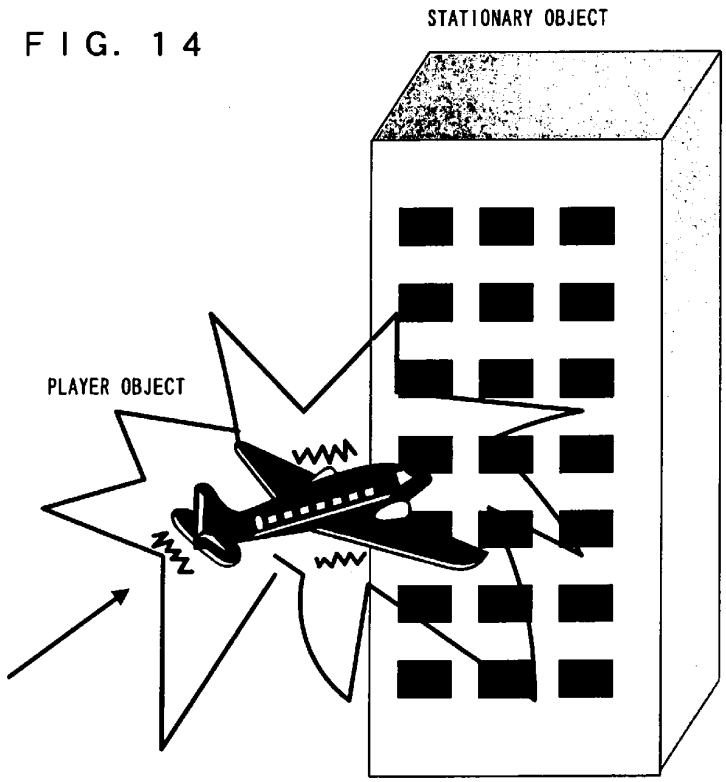


FIG. 15

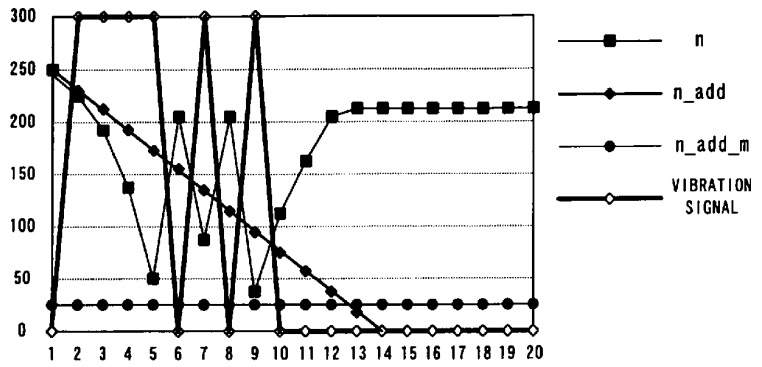

300
250
200
150
100
50
0
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
n
n_add
n_add_m
VIBRATION SIGNAL

FIG. 16

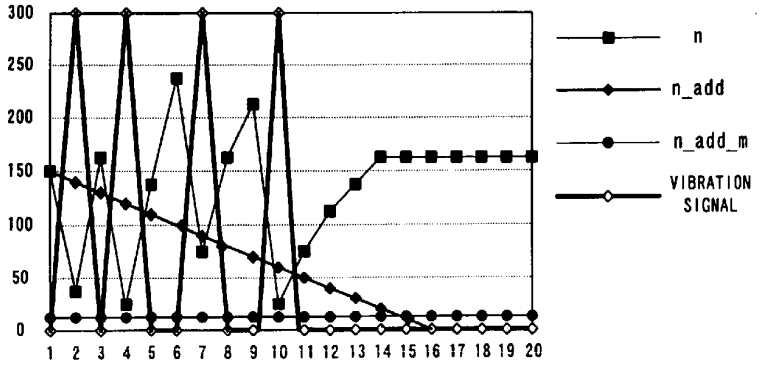
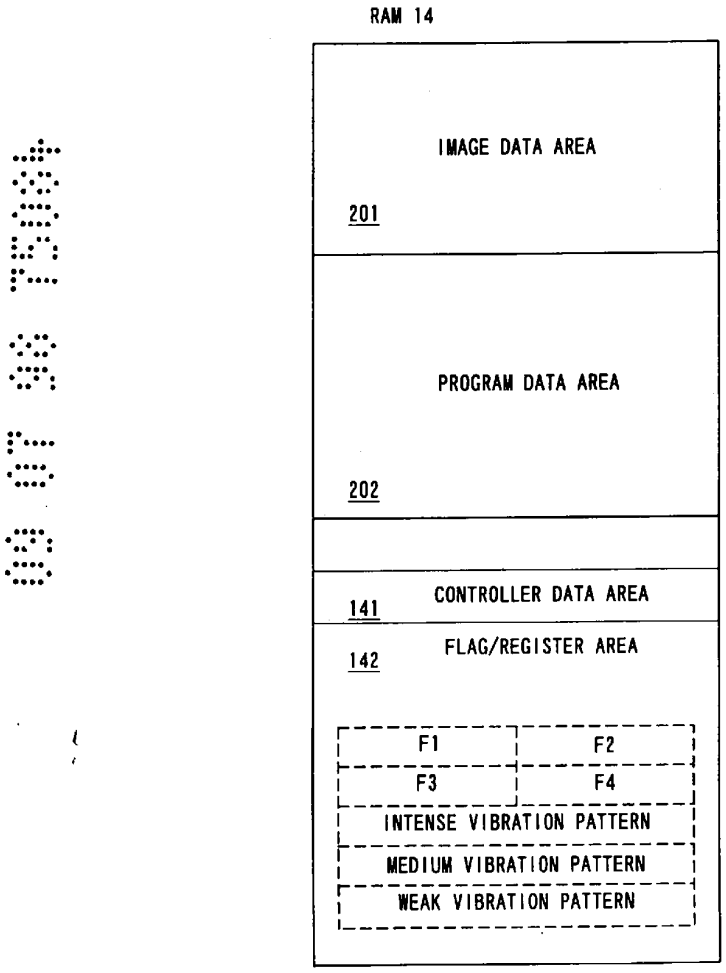

300
250
200
150
100
50
0
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
n
n_add
n_add_m
VIBRATION SIGNAL

FIG. 17


RAM 14
IMAGE DATA AREA
201
PROGRAM DATA AREA
202
CONTROLLER DATA AREA
141
FLAG/REGISTER AREA
142
F1
F2
F3
F4
INTENSE VIBRATION PATTERN
MEDIUM VIBRATION PATTERN
WEAK VIBRATION PATTERN