

ANTIALIASING USING MULTIPLE DISPLAY HEADS OF A GRAPHICS PROCESSOR

CROSS-REFERENCES TO RELATED APPLICATIONS

- 5 [0001] This application claims the benefit of U.S. Provisional Application No. 60/747,154, filed May 12, 2006, entitled "Antialiasing Using Multiple Display Heads of a Graphics Processor," and of commonly-assigned co-pending U.S. Patent Application No. 11/383,048, filed May 12, 2006, entitled "Distributed Antialiasing in a Multiprocessor Graphics System."

BACKGROUND OF THE INVENTION

- 10 [0002] The present invention relates in general to computer graphics, and in particular to antialiasing of image data using multiple display heads of a graphics processor.

- [0003] As is known in the art, computer-generated images are susceptible to various visual artifacts resulting from the finite sampling resolution used in converting the image data to an array of discrete color samples (pixels). Such artifacts, generally referred to as "aliasing,"
15 include jaggedness in smooth lines, irregularities in regular patterns, and so on.

- [0004] To reduce aliasing, color is often "oversampled," i.e., sampled at a number of sampling locations that exceeds the number of pixels making up the final (e.g., displayed or stored) image. For instance, an image might be sampled at twice or four times the number of pixels. Various types of oversampling are known in the art, including supersampling, in
20 which each sampling location is treated as a separate pixel, and multisampling, in which a single color value is computed for each primitive that covers at least part of the pixel, but coverage of the pixel by the primitive is determined at multiple locations.

- [0005] An antialiasing (AA) filter blends the multiple samples per pixel to determine a single color value. Conventionally, AA filters are applied either within the rendering pipeline
25 that generates pixels and stores them to a frame buffer or within the display pipeline that reads pixels from the frame buffer and delivers them to a display device.

BRIEF SUMMARY OF THE INVENTION

- [0006] Embodiments of the present invention provide systems and methods for exploiting
30 multiple display heads of a single graphics processor to perform antialiasing and other

processing tasks. In one embodiment, two display heads of the same graphics processor are coupled to each other in a master/slave configuration via a pixel transfer path. The "master" display head receives pixels from the "slave" display head in addition to its own pixels, and pixel selection logic in the master display head can blend the two pixels or select either one to the exclusion of the other. If the two pixels correspond to different sampling locations in the same image, the blended pixel is an AA-filtered pixel.

[0007] According to one aspect of the present invention, a graphics processing device includes a first display head, a second display head, and a pixel transfer path. The first display head is configured to generate a first output pixel and is disposed within an integrated circuit. The second display head, which is configured to generate a second output pixel, is also disposed within the integrated circuit. The second display head advantageously includes a first input path configured to receive an external pixel; a second input path configured to receive an internal pixel; a pixel combiner coupled to the first input path and the second input path and configured to blend the external pixel and the internal pixel to generate a blended pixel; and a selection circuit configured to select one of the external pixel, the internal pixel, or the blended pixel as a second output pixel. The pixel transfer path is configurable to deliver the first output pixel from the first display head to the first input path of the second display head such that the first output pixel is received by the first input path as the external pixel.

[0008] In some embodiments, the pixel transfer path is also disposed within the integrated circuit. In other embodiments, at least a portion of the pixel transfer path is external to the integrated circuit. For instance, the pixel transfer path may include a removable connector.

[0009] According to another aspect of the present invention, a graphics subsystem includes a graphics adapter having a pixel output connector and a pixel input connector. A graphics processor, which may be mounted on the graphics adapter, has a pixel output port communicably coupled to the pixel output connector and a pixel input port communicably coupled to the pixel input connector. The graphics subsystem also includes a removable connector unit adapted to connect the pixel output connector of the graphics adapter to the pixel input connector of the graphics adapter.

[0010] According to still another aspect of the present invention, a method of generating an image includes rendering a first set of input pixels and a second set of input pixels for the image using a rendering pipeline of a graphics processor. A first rendering operation used to

render the first set of input pixels differs in at least one respect from a second rendering operation used to render the second set of input pixels; for instance, the two rendering operations may differ with respect to a sampling pattern applied to each pixel or with respect to a viewport offset of the image being rendered. The first set of input pixels is delivered to a first display head of the graphics processor, and the second set of input pixels is delivered to a second display head of the graphics processor. The first set of input pixels is further delivered from the first display head to the second display head. In the second display head, corresponding pixels of the first set of input pixels and the second set of input pixels are blended to generate a set of output pixels.

10 [0011] The following detailed description together with the accompanying drawings will provide a better understanding of the nature and advantages of the present invention.

BRIEF DESCRIPTION OF THE DRAWINGS

15 [0012] FIG. 1 is a block diagram of a computer system according to an embodiment of the present invention;

[0013] FIG. 2 is a block diagram of a pixel output path in a graphics processing unit (GPU) usable in an embodiment of the present invention;

[0014] FIGS. 3A and 3B are block diagrams of pixel selection logic in a display head of a GPU usable in embodiments of the present invention;

20 [0015] FIG. 4A is a block diagram of a graphics subsystem with two GPUs according to an embodiment of the present invention;

[0016] FIG. 4B is a block diagram of a GPU showing two display heads coupled in a master/slave configuration according to an embodiment of the present invention;

25 [0017] FIGS. 5A-5C illustrate sampling patterns usable in some embodiments of the present invention;

[0018] FIG. 6 illustrates a graphics adapter implemented as a printed circuit card and configured with an external pixel transfer path according to an embodiment of the present invention; and

30 [0019] FIG. 7 is a block diagram of a GPU with an internal pixel transfer path according to an embodiment of the present invention.

DETAILED DESCRIPTION OF THE INVENTION

[0020] Embodiments of the present invention provide systems and methods for exploiting multiple display heads of a single graphics processor to perform antialiasing and other processing tasks. In one embodiment, two display heads of the same graphics processor are coupled to each other in a master/slave configuration via a pixel transfer path. The "master" display head receives pixels from the "slave" display head in addition to its own pixels, and pixel selection logic in the master display head can blend the two pixels or select either one to the exclusion of the other. If the two pixels correspond to different sampling locations in the same image, the blended pixel is an AA-filtered pixel.

System Overview

[0021] FIG. 1 is a block diagram of a computer system 100 according to an embodiment of the present invention. Computer system 100 includes a central processing unit (CPU) 102 and a system memory 104 communicating via a bus path that includes a memory bridge 105, e.g., a Northbridge chip. Memory bridge 105 is connected via a bus or other communication path 106 to an I/O (input/output) bridge 107, e.g., a Southbridge chip. I/O bridge 107 receives user input from one or more user input devices 108 (e.g., keyboard, mouse) and forwards the input to CPU 102 via bus 106 and memory bridge 105. Visual output is provided on a pixel based display device 110 (e.g., a conventional CRT or LCD based monitor) operating under control of a graphics subsystem 112 coupled to memory bridge 105 via a bus or other communication path 113. A system disk 114 is also connected to I/O bridge 107. A switch 116 provides connections between I/O bridge 107 and other components such as a network adapter 118 and various add-in cards 120, 121. Other components (not explicitly shown), including USB or other port connections, CD drives, DVD drives, and the like, may also be connected to I/O bridge 107. Communication paths among the various components may be implemented using protocols such as PCI (Peripheral Component Interconnect), PCI Express (PCI-E), AGP (Accelerated Graphics Port), HyperTransport, or any other bus or point-to-point protocol(s), and connections between different devices may use different protocols as is known in the art.

[0022] Graphics subsystem 112 includes a number N (one or more) of graphics processing units (GPUs) 122. (Herein, multiple instances of like objects are denoted with reference numbers identifying the object and parenthetical numbers identifying the instance where needed.) Each GPU 122 has an associated graphics memory 124. GPUs 122 and graphics

memories 124 may be implemented, e.g., using one or more integrated circuit devices such as programmable processors, application specific integrated circuits (ASICs), and memory devices. In some embodiments, GPUs 122 and graphics memories 124 are implemented in one or more expansion cards or other adapters that can be inserted into and removed from expansion slots (e.g., PCI-E slots) in system 100. Any number *N* of GPUs 122 may be used.

[0023] Each GPU 122 may be configured to perform various tasks related to generating pixel data (also referred to herein as "pixels") from graphics data supplied by CPU 102 and/or system memory 104 via memory bridge 105 and bus 113, interacting with a respective graphics memory 124 to store and update pixel data, and the like. For example, a GPU 122 may generate pixel data from 2-D or 3-D scene data provided by various programs executing on CPU 102. A GPU 122 may also write pixel data received via memory bridge 105 to graphics memory 124 with or without further processing. Each GPU 122 also includes a scanout module (also referred to herein as a display pipeline) that can be configured to deliver pixel data from graphics memory 124 an output port of the GPU 122 as described below. The output port might or might not be connected to a monitor or to another GPU 122.

[0024] For operations in a distributed rendering mode, one GPU (e.g., GPU 122(0)) is advantageously configured to deliver scanned-out pixels to another GPU (e.g., GPU 122(*N*-1)), which selects between internal pixels from its own display pipeline and external pixels received from GPU 122(0). More than two GPUs 122 can be interconnected in a "daisy-chain" fashion, such that a slave GPU 122 delivers its pixels to an intermediate GPU 122, which selects between its internal pixel and the external pixel from the slave, then forwards the selected pixel to another GPU, and so on until the ultimate master GPU (i.e., the GPU connected to a monitor) delivers the final selected pixels to a display device.

[0025] In some embodiments, GPUs 122 can be interconnected with each other in such a way that any GPU 122 can be slaved to any other GPU 122 by adjusting configuration settings of the GPUs 122 without any alteration of the physical connections. For instance, GPUs 122 can be connected in a unidirectional or bidirectional ring topology.

[0026] Various distributed rendering modes can be supported. For instance, in split-frame rendering, different GPUs 122 are assigned to render different portions of the same image; in alternate frame rendering, different GPUs 122 are assigned to render different images in a sequence of images to be displayed. A particular distributed rendering mode is not critical to the present invention.

[0027] In accordance with an embodiment of the present invention, GPUs 122 are also operable in an "externally distributed" AA mode. In this mode, pixel selection logic in the GPUs 122 blends internal and external pixels rather than selecting either pixel to the exclusion of the other. Where the internal and external pixels represent the same image with different sampling locations, the result of blending the pixels corresponds to an AA resolve operation (also referred to herein as an AA filter). In accordance with another embodiment of the present invention, one GPU 122 can be operated in an "internally distributed" AA mode. In this mode, the pixel selection logic in the GPU 122 blends pixels generated by two display heads of the same GPU 122. Examples of distributed AA modes (including internally and externally distributed modes) and associated pixel selection logic are described below.

[0028] In some embodiments, some or all of GPUs 122 may be also operable in an "independent rendering" mode in which different ones of GPUs 122 render images for different display devices; the images rendered by different GPUs 122 in an independent rendering mode might or might not be related to each other. It is to be understood that GPUs 122 may be configurable to operate in any of the above or other modes.

[0029] CPU 102 operates as the master processor of system 100, controlling and coordinating operations of other system components. In particular, CPU 102 issues commands that control the operation of GPU 122. In some embodiments, CPU 102 writes a stream of commands for GPU 122 to a command buffer, which may be in system memory 104, graphics memory 124, or another storage location accessible to both CPU 102 and GPU 122. GPU 122 reads the command stream from the command buffer and executes commands asynchronously with operation of CPU 102. The commands may include conventional rendering commands for generating images as well as general-purpose computation commands that enable applications executing on CPU 102 to leverage the computational power of GPU 122 for data processing that may be unrelated to image generation.

[0030] It will be appreciated that the system shown herein is illustrative and that variations and modifications are possible. The interconnection topology, including the number and arrangement of bridges, may be modified as desired. For instance, in some embodiments, system memory 104 is connected to CPU 102 directly rather than through a bridge, and other devices communicate with system memory 104 via memory bridge 105 and CPU 102. In other alternative topologies, graphics subsystem 112 is connected to I/O bridge 107 rather than to memory bridge 105. In still other embodiments, I/O bridge 107 and memory bridge

105 might be integrated into a single chip. The particular components shown herein are optional; for instance, any number of add-in cards or peripheral devices might be supported. In some embodiments, switch 116 is eliminated, and network adapter 118 and add-in cards 120, 121 connect directly to I/O bridge 107.

- 5 [0031] The connection of GPU(s) 122 to the rest of system 100 may also be varied. In some embodiments, graphics system 112 is implemented as one or more expansion, or add-in, cards that can be inserted into expansion slots of system 100. In other embodiments, a GPU is integrated on a single chip with a bus bridge, such as memory bridge 105 or I/O bridge 107.
- 10 [0032] Each GPU may be provided with any amount of local graphics memory, including no local memory, and may use local memory and system memory in any combination. For instance, in a unified memory architecture (UMA) embodiment, no dedicated graphics memory device is provided, and some or all of the GPUs use system memory exclusively or almost exclusively. In UMA embodiments, a GPU may be integrated into a bus bridge chip
- 15 or provided as a discrete chip with a high-speed bus (e.g., PCI-E) connecting the GPU to the bridge chip and system memory.

[0033] In addition, GPUs embodying aspects of the present invention may be incorporated into a variety of devices, including general purpose computer systems, video game consoles and other special purpose computer systems, DVD players, handheld devices such as mobile

20 phones or personal digital assistants, and so on.

GPU with Multiple Display Heads

- [0034] FIG. 2 is a block diagram of a pixel output path in a GPU 122 usable to practice the present invention. Although a multi-GPU graphics system is not required for some embodiments of the present invention, GPU 122 is advantageously configured to be usable in
- 25 such a system.

- [0035] In particular, as shown in FIG. 2, GPU 122 includes a display (or scanout) pipeline 202 coupled to a memory interface 204. Display pipeline 202 is also coupled to display heads 206a ("head A") and 206b ("head B"). GPU 122 has multiple output ports 210-213, including digital output ports 210, 211, and analog output ports 212, 213. GPU 122 also has
- 30 two multipurpose input/output (MIO) ports 214a ("MIO A") and 214b ("MIO B") that are configurable for various purposes, including communication with another GPU or with

another external digital device. Display heads 206a and 206b are each coupled to output ports 210-213 and MIO ports 214a, 214b via a crossbar 220.

[0036] Memory interface 204 is coupled to a memory (not shown in FIG. 2), e.g., graphics memory 124 of FIG. 1, that stores pixel data generated by GPU 122. Display pipeline 202 communicates with memory interface 204 to access the stored pixel data. Display pipeline 202 delivers the pixel data to either or both of display heads 206a, 206b. In some embodiments, display pipeline 202 may perform various processing operations on the pixel data before delivering it to display heads 206a, 206b, and pixel data destined for display head 206a might or might not be processed differently from pixel data destined for display head 206b. In addition, pixel data provided to display pipeline 202 for processing and delivery to display head 206a may be the same as or different from the pixel data provided to display pipeline 202 for processing and delivery to display head 206b. The particular configuration of display pipeline 202 and memory interface 204 is not critical to the present invention, and a detailed description is omitted.

[0037] Digital output ports 210, 211 may be of generally conventional design and may include circuits that modify the pixel data to conform to a digital output standard. For instance, in one embodiment, each of ports 210, 211 implements TMDS (Transition Minimized Differential Signaling) for a standard DVI (Digital Video Interface) connector. Similarly, analog output ports 212, 213 can be of generally conventional design and may include, e.g., a digital to analog converter conforming to any analog video standard, numerous examples of which are known in the art. It will be appreciated that the presence, absence, number, or nature of particular digital or analog output ports is not critical to the present invention.

[0038] MIO A port 214a and MIO B port 214b can be configured as output ports that drive pixel data produced by either of display heads 206a, 206b onto output lines of GPU 122. MIO A port 214a and MIO B port 214b can also be configured as input ports that deliver external pixel data to display head A 206a or display head B 206b. In some embodiments, MIO A port 214a and MIO B 214b are each independently configurable as either an input port or an output port. The configuration of MIO A port 214a and MIO B port 214b may be determined during system startup or dynamically modified at various times during system operation. For instance, each MIO port may include a control register that stores a value

specifying the port configuration, and a new value may be written to the register at system startup or at other times as desired.

[0039] Head A 206a and head B 206b are each coupled to output ports 210-213, as well as to MIO ports 214a, 214b via crossbar 220. In this embodiment, crossbar 220 is configurable to support any connection from head A 206a to any one of ports 210-213, 214a, or 214b and to simultaneously support any connection from head B 206b to any one of ports 210-213, 214a, or 214b that is not currently connected to head A 206a by crossbar 220. For instance, GPU 122 can simultaneously drive pixel data from heads 206a, 206b to two different monitors (e.g., via any two of digital output ports 210, 211 and/or analog output ports 212, 213). Alternatively, GPU 122 can simultaneously drive pixels to a monitor via one of output ports 210-213 and to another GPU via MIO A port 214a or MIO B port 214b. In some instances, one or both of display heads 206a, 206b may be idle, i.e., not delivering pixels to any output port.

[0040] MIO ports 214a, 214b can also be configured to receive pixel data from another one of GPUs 122 and to communicate the received pixel data into display heads 206a, 206b. Each GPU 122 also has pixel selection logic (described below) in each display head 206a, 206b to select an "external" pixel received from one of MIO ports 214a, 214b, an "internal" pixel received from its own display pipeline 202, or a combination of the internal and external pixels.

[0041] In some embodiments, crossbar 220 is configured at system startup; in other embodiments, crossbar 220 is dynamically configurable, so that the connections can be changed during system operation. Crossbar 220 may also be configurable to couple incoming pixel data received at one of MIO ports 214a, 214b to either of display heads 206a, 206b.

[0042] FIG. 3A is a block diagram of pixel selection logic 300 in display head 206a of a GPU 122 according to an embodiment of the present invention. It is to be understood that display head 206b may have pixel selection logic of similar design. In some embodiments, each display head 206a, 206b of GPU 122 has its own pixel selection logic 300.

[0043] Pixel selection logic 300 receives an internal pixel on a first path 302 from display pipeline 202 of FIG. 2. When MIO A port 214a of FIG. 2 (or in some embodiments, MIO B port 214b) is configured as an input port, pixel selection logic 300 also receives an external pixel on a second path 304.

[0044] The external pixel and the internal pixel are each propagated to a pixel combiner circuit 306, which blends the external pixel and the internal pixel to produce a blended pixel. Pixel combiner circuit 306 may be implemented, e.g., using conventional arithmetic logic circuits. In one embodiment, pixel combiner circuit 306 includes a first division circuit 308 that divides the internal pixel by one of a number of candidate divisors (e.g., 1, 2, 4, etc.); an addition circuit 310 that adds the internal pixel (after dividing) to the external pixel to produce a sum pixel; a selection circuit 312 that selects between the internal pixel and the sum pixel in response to a control signal (PSEL1); and a second division circuit 314 that divides the selected pixel by one of a number of candidate divisors (e.g., 1, 2, etc.), providing the result as a blended pixel on a path 316.

[0045] The external pixel on path 304 and the blended pixel on path 316 are presented to a selection circuit 318 (e.g., a multiplexer). In response to a control signal (PSEL2), selection circuit 318 selects either the internal pixel, the blended pixel, or the external pixel for delivery to an output path 320 that connects to crossbar 220 of FIG. 2.

[0046] The PSEL1 and PSEL2 signals are advantageously generated by control logic (not explicitly shown) in display head 206a. In some embodiments, this control logic, which may be of generally conventional design, is responsive to control information generated by a graphics driver program executing on CPU 102 of FIG. 1. Similar control information may also be used to control operation of pixel combiner 308, e.g., by selecting among candidate divisors for division circuits 308, 314 to produce a suitably weighted average for a particular application. Those skilled in the art with access to the present teachings will be able to implement suitable control logic for generating PSEL and pixel combiner control signals; accordingly, a detailed description of the control logic is omitted.

[0047] FIG. 3B is a block diagram of pixel selection logic 350 according to an alternative embodiment of the present invention. Pixel selection logic 350 is generally similar to pixel selection logic 300 of FIG. 3A but is advantageously designed with the capability to blend gamma-corrected pixels. Pixel selection logic 350 receives an internal pixel on a first path 352 from display pipeline 202 of FIG. 2. The internal pixel is propagated to a pixel combiner 358. When MIO A port 214a is configured as an input port, pixel selection logic 350 also receives an external pixel on a second path 354. The external pixel is also propagated to pixel combiner 358.

[0048] Pixel combiner 358 combines the internal and external pixels, providing a blended pixel as a result on signal path 360. Pixel combiner 358 may include a division circuit similar to division circuit 306 of FIG. 3A and/or an adder similar to adder circuit 308 of FIG. 3A. In addition (or instead), pixel combiner 358 may include arithmetic logic for blending gamma-corrected pixels received on paths 352 and 354. As is known in the art, gamma correction adjusts pixel values to a non-linear scale that will produce a linear intensity response in a display device, e.g., by converting a pixel value P to P^γ for a constant γ . (For typical display devices, the constant γ is around 2.0 to 2.5.) In one such embodiment, for $\gamma \approx 2.2$, a gamma-corrected output pixel P_o^γ can be computed using the equation:

10
$$P_o^\gamma = (4 P_i^\gamma + 4 P_e^\gamma + |P_i^\gamma - P_e^\gamma|)/4, \quad (\text{Eq. 1})$$

[0049] where P_i^γ and P_e^γ represent gamma-corrected pixels supplied on paths 352 and 354. Those skilled in the art will recognize that Eq. 1 provides an acceptable approximation using simpler hardware than computing an exact result would require. (For instance, multiplication and division by 4 can be implemented as bit shifts.) It will also be appreciated that other approximations may be substituted.

[0050] Referring again to FIG. 3B, a selection multiplexer 362 receives the internal pixel on path 352, the external pixel on path 354, and the blended pixel on path 360. In response to a pixel selection signal (PSEL), selection multiplexer 362 selects one of these candidate pixels for deliver to an output path 364 that connects to crossbar 220 of FIG. 2. Output path 364 may include a second division circuit 366, which may be similar to division circuit 314 of FIG. 3A. In an alternative embodiment, two selection multiplexers, similar to those shown in FIG. 3A, are used to select the output pixel, and division circuit 366 may be placed before the second selection multiplexer. Other selection circuit configurations may also be used.

[0051] In some embodiments, pixel combiner 350 is configurable to operate on either gamma-corrected pixels or non-gamma-corrected (linear) pixels, with simple addition being used when the pixels are not gamma corrected. The configuration is advantageously established by the graphics driver during a setup operation (e.g., at system startup). It is to be understood that gamma-corrected filters are not required; in some embodiments, any gamma correction may be applied after final pixel selection.

Distributed Antialiasing Using Multiple GPUs

[0052] GPU 122 of FIG. 2 with pixel selection logic 300 or pixel selection logic 350 is advantageously usable for a distributed antialiasing operation in a system with two or more GPUs 122. FIG.4A is a simplified block diagram of a graphics subsystem 400 having two GPUs 122(0) and 122(1) in a master/slave readout configuration. (For clarity, only the active ports and display heads are shown; in this example, only one display head per GPU is in use.) A slave GPU 122(1) has its MIO A port 214a(1) configured as an output port, while a master GPU 122(0) has its MIO A port 214a(0) configured as an input port. MIO A port 214a(1) is coupled to MIO A port 214a(0), allowing output pixels P_{oI} to flow from slave GPU 122(1) to master GPU 122(0).

[0053] Head A 206a(1) of slave GPU 122(1) forwards pixels P_{iI} provided by display pipeline 202(1) of slave GPU 122(1) as output pixels to MIO A port 214a(1). Output pixels P_{oI} from GPU 122(1) are received by MIO A port 214a(0) of master GPU 122(0), which forwards the pixels to display head A 206a(0). In head A 206a(0), pixel selection logic 300(0) (an instance of pixel selection logic 300 of FIG.3) operates to select internal pixels P_{i0} from display pipeline 202(0) of master GPU 122(0), external pixels P_{oI} originating in display pipeline 202(1) of slave GPU 122(1), or sums (e.g., $P_{i0} + P_{oI}$) supplied by addition circuit 308.

[0054] Head A 206a(0) of master GPU 122(0) delivers the selected pixels (P_{final}) to an output port, in this case digital output port 210(0). It will be appreciated that head A 206a(0) of GPU 122(0) could be configured to deliver pixel data to MIO B port 214b(0) (not explicitly shown in FIG.2C), which could be connected to an MIO port of a third GPU 122; the third GPU 122 would then be a master to GPU 122(0). Thus, any number of GPUs 122 may be connected in a daisy chain.

[0055] In accordance with an embodiment of the present invention, GPUs 122(0) and 122(1) can be used in an externally distributed antialiasing (AA) mode. In this mode, each GPU 122 renders the same image, with some variation in a viewing parameter or sampling parameter such that the sampling locations used by GPU 122(0) are different from the sampling locations used by GPU 122(1). For example, slightly different viewports or viewplane normals might be defined for the two GPUs 122, creating small offsets in the pixel boundaries of the two images. Alternatively, where the sampling location within a pixel is

configurable (e.g., by the graphics driver), each GPU 122 might be configured to use the same set of viewing parameters but a different sampling location within each pixel.

[0056] In a distributed AA mode, the external pixels P_{oI} and internal pixels P_{iI} received by pixel selection logic 300(0) in display head 206a(0) of GPU 122(0) correspond to different sampling locations for the same pixel of the final image. Averaging the internal and external pixels provides antialiasing at twice the display resolution. More specifically, selection multiplexer 310 is configured to select the pixel sum $P_{oI} + P_{iI}$ provided by adder circuit 308, and division circuit 316 is configured to divide the selected pixel sum by 2, so that the final pixel is $P_{final} = (P_{oI} + P_{iI})/2$. In this manner, pixel selection logic 300 can implement a 2x AA filter.

[0057] It should be noted that GPUs 122(0) and 122(1) can also be operated in other distributed rendering modes, including modes in which one of the internal pixel or the external pixel is selected to the exclusion of the other. The particular selection will depend on the specifics of the distributed rendering mode, e.g., whether different GPUs 122 are rendering different portions of the same frame or different successive frames, and is not relevant to the present invention. Further, where more than two GPUs 122 are present, a higher degree of antialiasing can be achieved. For example, any number of GPUs 122 can be connected in a daisy chain using the respective MIO A and MIO B ports thereof, and each GPU in the daisy chain can blend its own internal pixel with an external pixel received from an upstream GPU using appropriate weighting factors.

[0058] It will be appreciated that the display heads, pixel selection logic, and distributed AA operations described herein are illustrative and that variations and modifications are possible. For example, the division circuits referred to herein support division by a small number of discrete divisors. In other embodiments, the division circuits might support a larger number of divisors (including arbitrarily selected divisors) so that a broad range of antialiasing filters can be supported. Further, the division circuits may be placed at different locations from those described herein, and the number of division circuits may be modified. For instance, a division circuit might be placed on the external pixel path in addition to or instead of the internal pixel path.

[0059] The particular configuration of selection circuit 318 may also be modified. Those skilled in the art will recognize that any circuit element or combination of circuit elements

capable of controllably selecting among the internal pixel, the external pixel, and a blended pixel derived from both the internal and external pixel may be used as a selection circuit.

[0060] As used herein, a "pixel" refers generally to any representation of a color value sampled at some location within an image, or to a combination of such values (e.g., as produced by addition circuit 308 of FIG. 3). Rendering pipelines in the GPUs generate pixels at a nominal resolution (where resolution refers to the number of pixels in the image), which might or might not coincide with the resolution of the display device. In some embodiments, the display pipeline performs any needed up-filtering or down-filtering to transform the nominal resolution to the display resolution.

[0061] The labeling of MIO ports and display heads herein as "A" and "B" herein is solely for convenience of description. It is to be understood that any MIO port can be connected to any other MIO port, and either display head can drive either MIO port when that port is configured as an output port. In addition some GPUs may include more or fewer than two MIO ports and/or more or fewer than two display heads.

[0062] In general, any port or ports that enable one GPU to communicate pixel data with another GPU may be used as I/O ports to practice the present invention. In some embodiments, the MIO ports are also reconfigurable for purposes other than communicating with another GPU, as noted above. For instance, the MIO ports can be configured to communicate with various external devices such as TV encoders or the like; in some embodiments, DVO (Intel Corporation's Digital Video Output Interface) or other standards for video output can be supported. In some embodiments, the configuration of each MIO port is determined when a graphics adapter is assembled; at system startup, the adapter notifies the system as to the configuration of its MIO ports. In other embodiments, the MIO ports may be replaced with dedicated input or output ports.

[0063] Configuration of I/O ports, display heads, and other aspects of a graphics subsystem may be accomplished by a system setup unit configured to communicate with all of the graphics processors. In some embodiments, the system setup unit is implemented in a graphics driver program that executes on a CPU of a system that includes a multi-processor graphics subsystem. Any other suitable agent, including any combination of hardware and/or software components, may be used as a system setup unit.

Internally Distributed Antialiasing

[0064] In accordance with an embodiment of the present invention, the two display heads 206a, 206b of one GPU 122 may be coupled to each other in a master/slave configuration. In this configuration, GPU 122 can perform "internally distributed" AA filtering using pixel selection logic 300 in the display head (e.g., head A 206a) that is operating as the master.

[0065] FIG. 4 is a block diagram of a GPU 122 showing display head 206a coupled to display head 206b in a master/slave configuration according to an embodiment of the present invention. It is to be understood that GPU 122 of FIG. 4 may be identical to GPU 122 of FIG. 2; in FIG. 4, only the active I/O ports are shown, and crossbar 220 is not shown.

Display pipeline 202 in FIG. 4 is shown as having two parallel sections: display pipeline A 402a, which delivers pixels to display head 206a, and display pipeline B 402b, which delivers pixels to display head 206b. Display pipelines A 402a and B 402b may each be of generally conventional design, and each may be configured to perform various pixel processing operations; the two display pipelines 402a, 402b may perform the same operations or different operations, as desired.

[0066] MIO B port 214b is coupled, via a pixel transfer path 400, to MIO A port 214a of the same GPU 122. Pixel transfer path 400 transfers pixels produced by display head B 206b from MIO B port 214b to MIO A port 214a; MIO A port 214a delivers the pixels it receives to display head A 206a of GPU 122. Pixel transfer path 400 may be implemented using any suitable signal transfer techniques; examples are described below.

[0067] From the perspective of display head A 206a, the pixels received from display head B 206b are indistinguishable from pixels received from a different GPU. Thus, for instance, pixel selection logic 300 in display head A 206a can be operated to select, as an output pixel, any one of an "internal" pixel (P_A) originating from head A 206a, an "external" pixel (P_B) originating from head B 206b, or a blended pixel created from pixels P_A and P_B by pixel combiner circuit 308. (Pixels P_B are "external" to display head A 206a in the sense that, unlike pixels P_A , pixels P_B are not provided to display head A 206a by display pipeline A 402a.)

[0068] In this configuration, GPU 122 is usable to perform "internally distributed" AA, with the two display pipelines 402a, 402b supplying sample values that are blended by pixel selection logic 300 in display head A 206. In operation, a rendering pipeline (not explicitly shown) of GPU 122 renders two images of the same scene, with some variation in a viewing

parameter or sampling parameter such that the sampling locations used for the two images are different from each other. For example, slightly different viewports or viewplane normals might be defined for the two images, creating small offsets in the pixel boundaries of the two images. Alternatively, where the sampling location within a pixel is configurable (e.g., by the graphics driver), each image might be generated using the same set of viewing parameters but a different sampling location within each pixel.

[0069] One of the rendered images is stored in a frame buffer "A" 404 while the other is stored in a frame buffer "B" 406. Frame buffers A 404 and B 406 may be implemented in any memory device or devices, including on-chip memory in GPU 122, graphics memory 124 and/or system memory 104 of FIG. 1; the two frame buffers may be located in the same memory device or different devices as desired.

[0070] Display pipeline B 402b reads pixels from frame buffer B 406, performs various processing operations (which may be of a generally conventional nature) on the pixels, and forwards the resulting pixels P_B to display head B 206b. Display head B 206b has pixel selection logic 300 that operates to select pixels P_B ; those pixels are forwarded to MIO B port 214b via crossbar 220 (not explicitly shown in FIG. 4). Pixels P_B are transferred via pixel path 400 to MIO A port 214a of the same GPU 122, which forwards the pixels P_B to display head A 206a.

[0071] In parallel with this operation, display pipeline A 206a reads pixels from frame buffer A, performs various processing operations (which may be of a generally conventional nature) on the pixels, and forwards the resulting pixels P_A to display head A 206a. Display pipeline B 402b, display head B 206b, and pixel transfer path 400 are advantageously configured with appropriate timing so that pixel values P_A and P_B corresponding to the same screen pixel are delivered at the same time (e.g., in the same clock cycle) to pixel selection logic 300 of display head A 206a.

[0072] Within pixel combiner circuit 308, addition circuit 310 adds pixels P_A and P_B , multiplexer 312 selects the sum pixel, and division circuit 314 divides the sum by 2; thus, the blended pixel on path 316 is the average of pixels P_A and P_B . Multiplexer 318 selects the blended pixel as an output pixel P_{final} . Display head A 206a delivers the output pixel P_{final} to an output port (e.g., digital output port 210) for transmission to a display device.

[0073] It should be noted that because the rendering pipeline of GPU 122 renders each frame twice, the maximum frame rate for GPU 122 when operating in the internally

distributed AA mode described herein is generally lower than the maximum frame rate when operating in a non-AA mode. In some embodiments, the frame rate for this internally distributed AA mode is approximately half the frame rate for the non-AA mode. For real-time animation, as long as the frame rate for the internally distributed AA mode is
5 around 30 frames per second (or higher), the reduction in frame rate has little or no detrimental effect on the smoothness of the animation. Further, the image quality produced in a non-AA mode will generally be lower than image quality produced in an internally distributed AA mode; thus, internally distributed AA trades off a reduced frame rate for higher image quality.

10 [0074] It should also be noted that frame rates obtainable using the internally distributed AA mode described herein are comparable to frame rates obtainable using conventional AA techniques (e.g., filtering in the rendering pipeline and/or the display pipeline) in a single GPU. Conventional AA with a single GPU requires the GPU's rendering pipeline to generate a single image, but with multiple samples per pixel. Processing a larger number of samples
15 per pixel generally also decreases frame rate relative to non-AA modes in exchange for improved image quality. Depending on how the rendering of dual images is managed in the rendering pipeline, throughput of a GPU with internally distributed AA may be comparable to throughput of a GPU with conventional AA.

[0075] Higher-order AA filters may also be implemented, and such filters may employ a
20 combination of single-pipeline and internally distributed antialiasing operations. In one embodiment, display pipeline A 402a and display pipeline B 402b each include a filter-on-scanout (FOS) module (not explicitly shown) that implements an internal $N \times$ AA filter. More specifically, for each version of the image that is rendered, the rendering pipeline in GPU 122 generates a number N (e.g., 2, 4 or any other number larger than 1) of samples
25 per pixel, e.g., using conventional supersampling and/or multisampling techniques. The samples for one version of the image are stored in frame buffer A 404, while the samples for the other version of the image are stored in frame buffer B 406.

[0076] Display pipeline 402a receives all N samples for each pixel from frame buffer A. Within display pipeline 402a, a first filter-on-scanout (FOS) module (not explicitly shown in
30 FIG. 4) implements an $N \times$ AA filter, blending the N samples to produce a single color value per pixel. The color value determined by the FOS module is supplied (possibly after further processing) to display head A 206a as a pixel P_A .

[0077] Similarly, display pipeline 402b receives all N samples for each pixel from frame buffer B 406. Within display pipeline 402b, a second FOS module (also not explicitly shown in FIG. 4) implements an $N \times$ AA filter, blending the N samples to determine a single color value per pixel. The color value determined by the second FOS module is supplied (possibly after further processing) to display head B 206b as a pixel P_B .

[0078] Thus, the pixels P_A and P_B produced by display pipes 402a and 402b, respectively, can each be filtered pixels from an $N \times$ -oversampled image. As long as the sampling points used to populate frame buffer A 404 do not coincide with those used to populate frame buffer B 406, combining an $N \times$ AA filter in each display pipe 402a, 402b with the internally distributed AA filter technique described above results in a $(2N) \times$ AA filter. For instance, if the FOS module in each display pipe 402a, 402b provides a $4 \times$ AA filter, then GPU 122 can provide $8 \times$ AA.

[0079] A particular FOS module or AA filtering algorithm is not critical to the present invention, and conventional modules and algorithms may be used. Accordingly, a detailed description has been omitted. In some embodiments, the FOS modules in display pipelines 402a and 402b apply identical filter algorithms so that the resulting final image is not dependent on which version of the image is processed by a particular display pipeline. Further, $N \times$ AA filtering can be performed earlier in the image generation process. For instance, in one alternative embodiment, $N \times$ AA filtering might be performed within the rendering pipeline of GPU 122 using conventional techniques.

[0080] In some embodiments, the sampling points used to populate different frame buffers are selected such that no two sampling points coincide. For example, FIG. 5A illustrates a "grid" sampling pattern applied to a pixel 500. The pixel is sampled four times, at locations 501-504 (indicated by circles) within the pixel. FIG. 5B illustrates a "rotated grid" sampling pattern applied to pixel 500. The pixel is sampled four times, locations 511-514 (indicated by diamonds), which are different from locations 501-504.

[0081] In one embodiment, pixel data in frame buffer A 404 is generated using the grid sampling pattern of FIG. 5A while pixel data in frame buffer B 406 is generated using the rotated grid sampling pattern of FIG. 5B. The FOS module in display pipe A 402a filters the four sample values (501-504) to a single value P_A , while the FOS module in display pipe B 402b filters the four sample values (511-514) to a single value P_B . Pixel selection logic 300 in display head 206a blends the values P_A and P_B as described above to obtain a final pixel

value that corresponds to an average over all eight samples. This procedure provides the same antialiasing power as a single rendering process that samples each pixel with the eight-point pattern illustrated in FIG. 5C.

[0082] It will be appreciated that the internally distributed AA technique described herein is illustrative and that variations and modifications are possible. For instance, GPU 122 as described herein has exactly two display heads, each of which is capable of driving at most one output port; consequently, when both display heads are used for internally distributed antialiasing, GPU 122 can deliver at most one pixel stream to the display device(s).

However, embodiments of the present invention may be implemented in any GPU that has at least two display heads and suitable pixel selection logic and I/O ports. Where the GPU has more than two display heads, the GPU can support internally distributed AA and can also supply independent pixel streams to two or more display devices. Additionally, where the GPU has more than two display heads, it may be possible to connect all of the GPU's display heads together in a master/slave daisy chain to further increase the AA power of the GPU.

[0083] Further, GPU 122 as described herein has two MIO ports, both of which are used for internally distributed AA. In this embodiment, neither head A 206a nor head B 206b would be usable as a master or slave to any other GPU or display head. In other embodiments, the GPU may have additional MIO ports or the MIO ports may have an operating mode that allows one port to receive pixels and send pixels at the same time, allowing interconnectivity with other GPUs in combination with internally distributed AA. For example, where a third MIO port is present, that port might be configured as an input port to deliver external pixels from another GPU to display head B 206b or as an output port to deliver pixels produced by display head A 206a to another GPU. The other GPU in such an embodiment might or might not be configured to perform its own internally distributed AA filtering.

Pixel Transfer Path

[0084] Examples of pixel transfer path implementations according to embodiments of the present invention will now be described. As will become clear, the pixel transfer path may be external or internal to the GPU.

[0085] FIG. 6 illustrates a graphics adapter 600 implemented as a printed circuit card and configured with an external pixel transfer path according to an embodiment of the present invention. Graphics adapter 600 is implemented as an expansion card using a printed circuit

board (PCB) 602 that conforms to PCI-E or another interconnection standard. GPU 122 is mounted on PCB 602 and electrically coupled to a system connector 604 via wire traces (not shown) on PCB 602. System connector 604 is designed to be inserted into a PCI-E expansion slot (or any other type of expansion slot), enabling communication between GPU 122 and the rest of a computer system such as system 100 of FIG. 1. GPU 122 is also electrically coupled to a display output connector 606 via wire traces (not shown) on PCB 602. Display output connector 606 is advantageously coupled to one of digital output ports 210, 211 or analog output ports 212, 213 of GPU 122 (see FIG. 2). In some embodiments, PCB 602 may provide multiple display output connectors 606, each coupled to a different one of output ports 210-213, as is known in the art.

[0086] PCB 602 also includes two graphics edge connectors 614a, 614b, which can be of identical design. Graphics edge connector 614a connects to MIO A port 214a of GPU 122 via wire traces 616 while graphics edge connector 614b connects to MIO B port 214b of GPU 122 via wire traces 618. Each graphics edge connector 614a, 614b is configured for electrical and mechanical connection to a removable interconnect device. In some embodiments, graphics edge connectors 614a and 614b are of identical configuration, allowing them to be used interchangeably.

[0087] Graphics adapter 600 in one embodiment is designed for use in distributed rendering systems in which two or more GPUs cooperate to perform different portions of a rendering task. Such systems may be operated, e.g., in a split-frame mode in which each GPU renders a different part of an image, in an alternate-frame mode in which each GPU renders different images in a sequence of images, or in a distributed antialiasing mode. In each of these modes, one GPU (the master) receives pixels from another GPU (the slave), and pixel selection logic 300 in the master GPU selects a pixel for display as described above. GPUs on different graphics adapters 600 are advantageously connected via respective graphics edge connectors 614a, 614b using a suitable interconnection device.

[0088] In an embodiment of the present invention, a removable interconnection device 620 is constructed and shaped such that it can connect graphics edge connectors 614a and 614b of the same graphics adapter 600, as shown in FIG. 6. Interconnect device 620 can be, e.g., a ribbon cable or a PCB with wire traces printed along its length, with receptacles at either end for receiving a graphics edge connector 614a, 614b, allowing the two graphics edge connectors 614a, 614b to be connected to each other.

[0089] In this embodiment, interconnection device 620 exploits the timing characteristics of the distributed-rendering system supported by graphics adapter 600 to establish a pixel transfer path 400 (FIG. 4). More specifically, in a distributed rendering configuration, a pixel transfer path from MIO B port 214b of GPU 122 to an MIO A port (or MIO B port) of a GPU on a different graphics adapter 600 has a characteristic transfer time, resulting from the lengths of wire traces 618 and/or 616 and the interconnect device between the two adapters, as well as any electronic components (FIFOs, latches, etc.) that may be included along any segment of the transfer path. In distributed rendering operations, the transfer of pixels from the display head of the slave GPU to the display head of the master GPU is advantageously coordinated such that the pixels from the slave GPU and master GPU arrive at the display head of the master GPU at substantially the same time (e.g., during the same clock cycle).

[0090] As long as interconnection device 620 provides a transmission time matching that of a distributed-rendering interconnection device that connects different GPUs, the pixel transfer path provided by interconnection device 620 delivers signals to MIO A port 214a with the correct timing. Thus, implementation of internally distributed AA using an external interconnection device requires no internal modifications to a GPU 122 or an adapter card 600 that was originally designed for distributed rendering.

[0091] It will be appreciated that the graphics adapters and interconnection devices described herein are illustrative and that variations and modifications are possible. The shape, layout, and material composition of the adapters and interconnection devices may be modified from those shown and described herein, and any communication protocol may be implemented for transferring data between MIO ports.

[0092] In one alternative embodiment, interconnection device 620 might be implemented as part of PCB 602, e.g., using wire traces to connect path 618 to path 616. In this embodiment, control devices (e.g., a removable jumper or a driver-controlled switch) are advantageously used to enable or disable data transfers from path 618 to path 616 or vice versa.

[0093] It should also be noted that in some embodiments the presence of interconnection device 620 or other external connection between two MIO ports of the same GPU does not automatically enable internally distributed AA. As described above, the operation of pixel selection logic 300 determines whether internally distributed AA is performed; operation of pixel selection logic 300 is controlled via the graphics driver.

[0094] In another alternative embodiment, the pixel transfer path used for internally distributed AA is built within the GPU. FIG. 7 is a block diagram of a GPU 700 according to one such embodiment of the present invention. GPU 700 is generally similar to GPU 122 of FIG. 4, and like reference numbers have been used to identify corresponding components.

5 Unlike GPU 122, GPU 700 includes an internal pixel transfer path connecting an output path 702 of display head B 206b to an external pixel input path 704 of display head A 206a.

[0095] In this embodiment, the pixel transfer path includes a selection unit (e.g., a multiplexer) 706 that selects between the pixel from display head B 206b and a pixel received on a path 708 from one of the MIO ports, e.g., MIO A port 214a, via crossbar 220. The
10 selected pixel is provided to the external pixel input path 704 of display head A 206a.

[0096] Selection unit 706 operates in response to a control signal (not explicitly shown). The control signal configures selection unit 706 to select the pixel on path 702 in the event that GPU 700 is operating in internally distributed AA mode and to select the pixel on path 708 in the event that display head A 206a of GPU 700 is operating as a master to another
15 GPU. This control signal may be generated in response to commands issued by the graphics driver, enabling a user (or application developer) to enable or disable internally distributed AA through an appropriate software interface without having to access the graphics hardware.

[0097] It should be noted that in this embodiment, path 702 from display head B 206b to the selection circuit 706 may include FIFOs, latches, and other timing control devices so that
20 pixels from display head B 206b reach selection circuit 706 with the same timing as would pixels arriving from an external GPU in a distributed rendering mode. Where this is the case, the operational timing of display head B 206b and display head A 206a is independent of whether the GPU is in a distributed rendering mode or an internally distributed AA mode.

25 **[0098]** An internal pixel transfer path, while requiring modifications to the GPU, does not require use of any of the GPU's I/O ports. Thus, for instance, display head A 206a of GPU 700 can be slaved to a display head in another GPU, or display head B 206b of GPU 700 can be master to a display head in another GPU while GPU 700 continues to perform internally distributed AA filtering.

30 **[0099]** It will be appreciated that the internal pixel transfer path described herein is illustrative and that variations and modifications are possible. For instance, a "reverse" pixel

transfer path (from display head A 206a to display head B 206b) might be provided in addition to the path shown.

Further embodiments

[0100] As described above, embodiments of the present invention provide distributed
5 antialiasing techniques for multi-GPU systems as well as a single GPU with the capability of using readout techniques and components generally associated with distributed rendering across multiple GPUs to generate AA-filtered images. Via a suitable graphics driver interface, an end user of an appropriately configured GPU can elect to enable internally distributed AA for any graphics program, regardless of the AA (or lack thereof) provided in
10 the program itself. Where the program provides AA, internally distributed AA as described herein can be used to increase (e.g., double) the AA resolution.

[0101] While the invention has been described with respect to specific embodiments, one skilled in the art will recognize that numerous modifications are possible. For instance, although the invention has been described with reference to AA filtering, the coupling
15 between display heads of a single GPU or between display heads of multiple GPUs described herein may be used in other ways.

[0102] In one alternative embodiment, distributed filtering can be used to generate stereo anaglyphs. As is known in the art, a stereo anaglyph overlays a left-eye view and a right-eye view of a scene to produce a single image. Typically, different color filters are applied to the
20 left-eye pixels and the right-eye pixels; for instance, the right-eye pixels may be filtered with a red-pass filter while the left-eye pixels are filtered using a blue/green-pass filter. Due to a viewport or viewpoint offset between the left-eye and right-eye views, the left-eye pixel and right-eye pixel corresponding to the same point in the scene are in different places in the anaglyph. Thus, to the naked eye, an anaglyph appears as a double image with distorted
25 colors. To view the image properly, a viewer dons special glasses with a left lens that filters out the colors used for right-eye pixels and a right lens that filters out the colors used for left-eye pixels.

[0103] Referring to FIG. 4A, a rendering pipeline (not shown) of GPU 122(0) may be used to generate a right-eye view of a scene while a rendering pipeline (not shown) of GPU 122(1)
30 generates a left-eye view of the same scene. Known techniques may be used to determine rendering parameters for the right-eye and left-eye views.

[0104] The right-eye pixels P_{il} and left-eye pixels P_{io} are advantageously color-filtered, either in the respective rendering pipelines of GPUs 122(0) and 122(1) or in respective display pipelines 202(0) and 202(1). In one embodiment, pixel color is specified using distinct red, green, and blue color components. Right-eye pixels can be filtered, e.g., by
5 reducing the red component to zero and leaving the green and blue components unaltered; similarly, left-eye pixels can be filtered by reducing the green and blue components to zero and leaving the red component unaltered.

[0105] Right-eye pixels P_{il} are delivered to display head A 206a(1) of GPU 122(1). Display head 206(a) forwards pixels P_{il} to MIO A port 214a(1), which delivers them, as
10 pixels P_{ol} , to MIO A port 214a(0) of GPU 122(0). Display head 206a(0) thus receives the right-eye pixels as external pixels.

[0106] Left-eye pixels P_{io} are delivered to display head 206a(0) as internal pixels. It should be noted that, because of the offset between the viewports or viewpoints used to generate the right-eye and left-eye views, corresponding left-eye and right-eye pixels processed by the
15 pixel selection logic 300 are pixels at the same location in the anaglyph frame; in general, corresponding left-eye and right-eye pixels represent different locations in the scene.

[0107] In one embodiment display head 206a(0) includes pixel selection logic 300 of FIG.3. For generating anaglyphs, division circuits 306 and 314 are both set to select a divisor of 1. Addition circuit 308 adds the left-eye pixel P_{io} and right-eye pixel P_{il} to produce a sum
20 pixel on path 310 that, as a result of the earlier color filtering, has the red component of the left-eye pixel and the blue and green components of the right-eye pixel. Selection multiplexer 312 selects the sum pixel from path 310, and selection multiplexer 316 selects the blended pixel on path 315 as the output pixel.

[0108] In other embodiments, color filters prior to pixel selection logic 300 are not used.
25 For example, selection multiplexers 312 and/or 316 can be configured such that the selection can be controlled independently for each color component. In one such embodiment, selection multiplexer 312 passes through all color components of left-eye pixel P_{io} from path 302, while selection multiplexer 316 passes through the red component of left-eye pixel P_{io} and the blue and green components of right-eye pixel P_{il} . The result is an output pixel on
30 path 318 that has the red component of the left-eye pixel and the blue and green components of the right-eye pixel.

[0109] Those skilled in the art will appreciate that more than two GPUs can be used for anaglyph rendering. In an embodiment with four GPUs (e.g., FIG.4B), two GPUs can be used to generate the right-eye view while the other two GPUs are used to generate the left-eye view. The two GPUs generating each view can employ distributed antialiasing techniques as described above to improve the image quality of the anaglyph.

[0110] Stereo anaglyphs can also be rendered using internally distributed filtering. Referring to FIG. 4B, the rendering pipeline (not shown) of GPU 122 can generate both views, storing the left-eye view in frame buffer A and the right-eye view in frame buffer B (or vice versa). Display pipeline B 402b and display head B 206b deliver the right-eye pixels to display head A 206a as pixels P_B while display pipeline A 402a delivers the left-eye pixels to display head A 206a as pixels P_A . In display head A 206a, pixel combiner 308 (FIG. 3) blends the pixels appropriately to produce the anaglyph.

[0111] Distributed filtering can also be used to generate transitional effects such as fade-in, fade-out, or dissolve. For instance, in the internally distributed case, frame buffer B may store an image that is fading out while frame buffer A stores an image that is fading in. At each frame, pixel combiner 308 adjusts the relative weights of the pixels from frame buffer A and frame buffer B, so that the image from frame buffer A gradually increases to full intensity while the image in frame buffer B fades to zero intensity. (If the image in frame buffer B is a solid color field, the effect is a fade-in; if the image in frame buffer A is a solid color, the effect is a fade-out.) The smoothness of the transition depends in part on the number of different weighted averages of pixels P_A and P_B pixel combiner 308 is capable of forming, which is a matter of design choice. Multiple GPUs with externally distributed filtering can also be used to achieve a similar effect.

[0112] In another embodiment, such transitional effects can be achieved using distributed filtering in combination with a lookup table in each display head. As is known in the art, a display head often includes a lookup table that converts the internal pixel representation to a color intensity value appropriate for a display device, and different values can be loaded into the lookup table can be reloaded from time to time. Fade out (or fade in) can be achieved by reducing (or increasing) the color intensity of the values in the lookup table from one frame to the next. Thus, to dissolve from an image in frame buffer B to an image in frame buffer A, conventional fade-out lookup tables could be applied in display head B while conventional

fade-in lookup tables are applied in display head A. Pixel combiner 308 would combine the two images with constant (e.g., equal) weights to create the dissolve effect.

[0113] In other embodiments, pixel transfer between display heads of the same GPU is used to implement display features that do not involve blending. For instance, pixel transfer
5 between display heads can be used to control an LCD overdrive (also referred to in the art as "LCD feed-forward" or "response time compensation" (RTC)) function. As is known in the art, an LCD screen can be made to respond faster if the signals driving the pixels are adjusted from frame to frame based in part on the desired new intensity and in part on the difference between the desired new intensity and the previous intensity.

10 **[0114]** To implement an LCD overdrive function, frame buffer A can be used to store pixels of a new image while frame buffer B stores pixels of a previous image. Display head B delivers the previous pixel values to display head A, and pixel combiner 308 of display head A can be configured to compute an overdrive value based on the new value and the previous value, e.g., using conventional techniques for computing an LCD overdrive signal.

15 **[0115]** A pixel transfer between display heads of a GPU can also be used for generating composite images. For instance, frame buffer B may contain pixels for an overlay image to be overlaid on part of an image stored in frame buffer A. Display head B delivers overlay pixels to display head A, and pixel selection logic 300 in display head A selects the internal pixel except in the overlay region, where the external pixel is selected.

20 **[0116]** Thus, although the invention has been described with respect to specific embodiments, it will be appreciated that the invention is intended to cover all modifications and equivalents within the scope of the following claims.

WHAT IS CLAIMED IS:

1 1. A graphics processing device comprising:
2 a first display head configured to generate a first output pixel, the first display
3 head being disposed within an integrated circuit;
4 a second display head configured to generate a second output pixel, the second
5 display head being disposed within the integrated circuit, the second display head including:
6 a first input path configured to receive an external pixel;
7 a second input path configured to receive an internal pixel;
8 a pixel combiner coupled to the first input path and the second input
9 path and configured to blend the external pixel and the internal pixel to generate a
10 blended pixel; and
11 a selection circuit configured to select one of the external pixel, the
12 internal pixel, or the blended pixel as a second output pixel; and
13 a pixel transfer path configurable to deliver the first output pixel from the first
14 display head to the first input path of the second display head such that the first output pixel
15 is received by the first input path as the external pixel.

1 2. The graphics processing device of claim 1 wherein the pixel transfer
2 path is disposed within the integrated circuit.

1 3. The graphics processing device of claim 1 wherein at least a portion of
2 the pixel transfer path is external to the integrated circuit.

1 4. The graphics processing device of claim 3 wherein the pixel transfer
2 path includes a removable connector.

1 5. The graphics processing device of claim 1 further comprising:
2 a first display pipeline configured to generate a first pixel and to deliver the
3 first pixel to the first display head; and
4 a second display pipeline configured to generate a second pixel and to deliver
5 the second pixel to the second display head.

1 6. The graphics processing device of claim 5 further comprising:

2 a rendering pipeline configured to generate first and second versions of a
3 frame to be displayed, the first and second versions being different from each other in at least
4 one respect,

5 wherein the first display pipeline is configured to generate the first pixel based
6 on the first version of the frame, the second display pipeline is configured to generate the
7 second pixel based on the second version of the frame, and the selection circuit in the second
8 display head is configured to select the blended pixel as the second output pixel.

1 7. A method of generating an image, the method comprising:
2 rendering a first set of input pixels and a second set of input pixels for the
3 image using a rendering pipeline of a graphics processor, wherein a first rendering operation
4 used to render the first set of input pixels differs in at least one respect from a second
5 rendering operation used to render the second set of input pixels;
6 delivering the first set of input pixels to a first display head of the graphics
7 processor;
8 delivering the second set of input pixels to a second display head of the
9 graphics processor;
10 delivering the first set of input pixels from the first display head to the second
11 display head; and
12 in the second display head, blending corresponding pixels of the first set of
13 input pixels and the second set of input pixels to generate a set of output pixels.

1 8. A display head for a graphics processor comprising:
2 a first input path configured to propagate a first pixel generated by a first
3 graphics processor;
4 a second input path configured to propagate a second pixel generated by a
5 second graphics processor;
6 a pixel combiner coupled to the first input path and the second input path and
7 configured to blend the first pixel and the second pixel to generate a blended pixel; and
8 a selection circuit configured to select one of the first pixel, the second pixel,
9 or the blended pixel as an output pixel.

1 9. The graphics processor of claim 8 wherein the display head further
2 includes:
3 a division circuit configured to divide the blended pixel by a divisor.

1 10. The graphics processor of claim 8 wherein the pixel combiner includes
2 a division circuit configured to divide the first pixel by a divisor prior to blending the first
3 pixel and the second pixel.

1 11. The graphics processor of claim 8 wherein the pixel combiner is
2 configured to generate the blended pixel by adding the first pixel and the second pixel.

1 12. The graphics processor of claim 8 wherein the first pixel and the
2 second pixel are gamma-corrected pixels and wherein the pixel combiner is configured to
3 generate the blended pixel by computing a gamma-corrected blend of the first pixel and the
4 second pixel.

1 13. A method of generating an image, the method comprising:
2 rendering a first set of input pixels for the image using a first graphics
3 processor;

4 rendering a second set of input pixels for the image using a second graphics
5 processor, wherein respective rendering operations performed by the first and second
6 graphics processors differ in at least one respect;

7 delivering the first set of input pixels and the second set of input pixels to a
8 first display head;

9 in the first display head, blending corresponding pixels of the first set of input
10 pixels and the second set of input pixels to generate a first set of output pixels.