(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
1 August 2013 (01.08.2013)

WIPO | PCT

(10) International Publication Number
WO 2013/112630 A1

(51) International Patent Classification:
G06F 3/00 (2006.01)

(21) International Application Number:
PCT/US2013/022805

(22) International Filing Date:
23 January 2013 (23.01.2013)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
61/589,824 23 January 2012 (23.01.2012) US
13/625,801 24 September 2012 (24.09.2012) US

(71) Applicant *(for all designated States except US)*: **GOOGLE INC.** [US/US]; 1600 Amphitheatre Parkway, Mountain View, CA 94043 (US).

(72) Inventors; and
(71) Applicants *(for US only)*: **QIN, Min** [US/US]; 19500 Pruneridge Ave., #6101, Cupertino, CA 95014 (US). **KLOBA, Grace** [US/US]; 938 Lorne Way, Sunnyvale, CA 94087 (US). **REN, Huan** [US/US]; 2008 Klee Place, Davis, CA 95618 (US).

(74) Agent: **MICKELSEN, Andrew, D.**; Mcdermott Will & Emery LlP, 4 Park Plaza, Suite 1700, Irvine, CA 92614 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— *with international search report (Art. 21(3))*

(54) **Title**: RENDERING CONTENT ON COMPUTING SYSTEMS

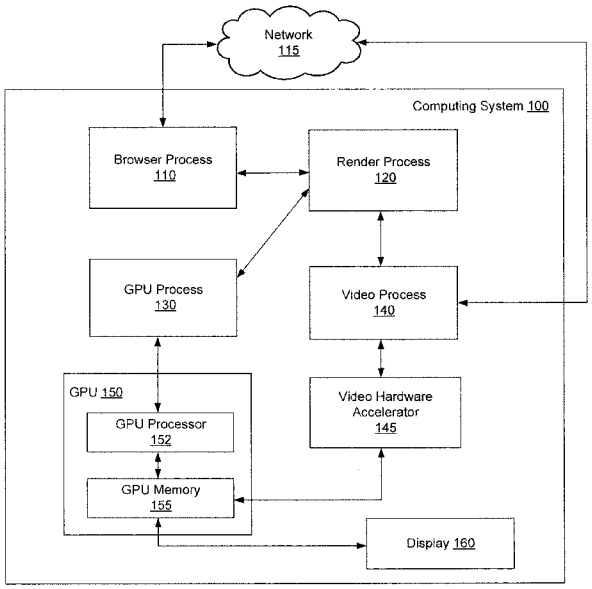

FIG. 1

(57) **Abstract**: A computer-implemented method for rendering a web page including web content and video content is disclosed according to an aspect of the subject technology. The method comprises retrieving the web content from a network using a browser process implemented on one or more processors, and rendering the retrieved web content into rendered web content using a render process implemented on the one or more processors. The method also comprises retrieving the video content from the network using a video process implemented on the one or more processors, and directing the retrieved video content to a video hardware accelerator, wherein the video hardware accelerator renders the video content into a rendered video frame. The method further comprises instructing a graphics processing unit (GPU) to composite the rendered web content and the rendered video frame into a composite frame for display to a user.

WO 2013/112630 A1

RENDERING CONTENT ON COMPUTING SYSTEMS

CROSS-RFERENCE TO RELATED APPLICATION

[0001] The present application claims the benefit of U.S. Provisional Patent Application Serial No. 61/589,824, entitled "Rendering Content on Computing Systems," filed on January 23, 2012, which is hereby incorporated by reference in its entirety for all purposes.

FIELD

[0002] The subject disclosure generally relates to computing systems, and, in particular, to rendering content on computing systems.

BACKGROUND

[0003] A browser may be used to retrieve and view a web page from a network (e.g., the Internet). The web page may include text, images, video content and/or other forms of web content. The browser may include a renderer that is configured to render the web content for display to the user. The renderer may do this by parsing and interpreting markup language code describing how various web content of the web page are to be rendered and laid out. The renderer may render video content using video processing software running on a general-purpose processor. However, this may result in slow and low-performance rendering of the video content, negatively impacting the user's browsing experience.

SUMMARY

[0004] A computer-implemented method for rendering a web page including web content and video content is disclosed according to an aspect of the subject technology. The method comprises retrieving the web content from a network using a browser process implemented on one or more processors, and rendering the retrieved web content into rendered web content using a render process implemented on the one or more processors. The method also comprises retrieving the video content from the network using a video process implemented on the one or more processors, and directing the retrieved video content to a video hardware accelerator, wherein the video hardware accelerator renders the video content into a rendered video frame.

The method further comprises instructing a graphics processing unit (GPU) to composite the rendered web content and the rendered video frame into a composite frame for display to a user.

[0005] A system for rendering a web page including web content and video content is disclosed according to an aspect of the subject technology. The system comprises one or more processors, a video hardware accelerator, a graphics processing unit (GPU), and a machine-readable medium comprising instructions stored therein, which when executed by the one or more processors, cause the one or more processors to perform operations. The operations comprise retrieving the web content from a network, rendering the retrieved web content into rendered web content, retrieving the video content from the network, and directing the retrieved video content to the video hardware accelerator, wherein the video hardware accelerator renders the video content into a rendered video frame and writes the rendered video frame to a memory space in the GPU. The operations also comprise instructing the GPU to composite the rendered video frame in the memory space and the rendered web content into a composite frame for display to a user.

[0006] A machine-readable medium is disclosed according to an aspect of the subject technology. The machine-readable medium comprises instructions stored therein, which when executed by a machine, cause the machine to perform operations for rendering a web page including web content and video content. The operations comprise retrieving the web content from a network, rendering the retrieved web content into rendered web content, storing the rendered web content in a memory, retrieving the video content from the network, and directing the retrieved video content to a video hardware accelerator, wherein the video hardware accelerator renders the video content into a rendered video frame and writes the rendered video frame to a memory space in a graphics processing unit (GPU). The operations also comprise transferring the rendered web content from the memory to the GPU, and instructing the GPU to composite the rendered web content transferred from the memory and the rendered video frame in the memory space in the GPU into a composite frame for display to a user.

[0007] It is understood that other configurations of the subject technology will become readily apparent to those skilled in the art from the following detailed description, wherein various configurations of the subject technology are shown and described by way of illustration. As will

be realized, the subject technology is capable of other and different configurations and its several details are capable of modification in various other respects, all without departing from the scope of the subject technology. Accordingly, the drawings and detailed description are to be regarded as illustrative in nature and not as restrictive.

BRIEF DESCRIPTION OF THE DRAWINGS

[0008] Certain features of the subject technology are set forth in the appended claims. However, for purpose of explanation, several embodiments of the subject technology are set forth in the following figures.

[0009] FIG. 1 is a conceptual block diagram of a computing system implementing a multi-process architecture for rendering content according to an aspect of the subject technology.

[0010] FIG. 2 shows a method for rendering content according to an aspect of the subject technology.

[0011] FIG. 3 conceptually illustrates an electronic system with which some implementations of the subject technology are implemented.

DETAILED DESCRIPTION

[0012] The detailed description set forth below is intended as a description of various configurations of the subject technology and is not intended to represent the only configurations in which the subject technology may be practiced. The appended drawings are incorporated herein and constitute a part of the detailed description. The detailed description includes specific details for the purpose of providing a thorough understanding of the subject technology. However, it will be clear and apparent to those skilled in the art that the subject technology is not limited to the specific details set forth herein and may be practiced without these specific details. In some instances, well-known structures and components are shown in block diagram form in order to avoid obscuring the concepts of the subject technology.

[0013] The subject technology quickly and efficiently renders video content in a web page by creating a separate video process to handle rendering of the video content. The video process may be implemented as a process in a multi-process architecture. The video process fetches the

video content from the network and feeds the received video content to a video hardware accelerator, which provides fast and high performance video processing. In addition, the video hardware accelerator may write the rendered video content directly to a hardware memory, which is readily accessible by a graphics processing unit (GPU) for hardware-accelerated compositing of the rendered video content with other rendered web content for display to the user.

[0014] FIG. 1 shows an example of a computing system 100 comprising multiple processes in a multi-process architecture according to an aspect of the subject technology. The multiple processes may include a browser process 110, a render process 120, a graphics processing unit (GPU) process 130, and a video process 140. The various processes may be implemented on one or more processors and may communicate with one another via inter-process communication (IPC).

[0015] The multiple processes increase robustness of a web browser. This is because, if one of the processes crashes, then the crash may be isolated to that process so that the other processes are not affected. In other words, the entire web browser is not brought down when one of the processes crashes (e.g., the render process 120).

[0016] In one aspect, the various processes may share data via a shared memory (not shown in FIG. 1). In this aspect, one of the processes may write data to the shared memory and share the data with another process by sending a pointer to the other process indicating where the data is stored in the shared memory. The shared memory may comprise an internal memory of the one or more processors implementing the processes and/or an external memory.

[0017] The computing system 100 also comprises a graphics processing unit (GPU) 150 for performing drawing and compositing operations, as discussed further below. The GPU 150 may include a GPU processor 152 and a GPU memory 155. The GPU processor 152 may be configured to provide accelerated compositing of various rendered web content into a composite frame for display on a display 160, as discussed further below. The GPU memory 155 may be used to temporarily store rendered content for compositing and the composite frame.

[0018] The computing system 100 also comprises a video hardware accelerator 145 for hardware-accelerated processing (e.g., video decoding) of video content. The video hardware accelerator 145 may support video decoding for various video formats including H.265, VP8, MPEG-4, VC-1 and/or other video formats. An advantage of using the video hardware accelerator 145 for video processing is that the video hardware accelerator 145 can provide higher speed and performance for video processing compared with software running on a general-purpose processor (e.g., a processor on which the processes run). Although the video hardware accelerator 145 is shown separately from the GPU 150 in FIG. 1, it is to be understood that the video hardware accelerator 145 may be part of the GPU 150.

[0019] In one aspect, the browser process 110 is configured to manage the browser user interface. The browser process 110 may also be configured to request and receive a web page from a network 115 (e.g., when the user clicks on a link), and send the received web page to the render process 120 for rendering. The web page may include Hyper Text Markup Language (HTML) code or other markup language code describing how various web content (e.g., text, image, video) of the web page are to be rendered and the laid out. The HTML code may also include one or more links (e.g., uniform resource locators (URLs)) to web content for the web page.

[0020] The browser process 110 may also determine the position and size of the browser window on the display 160, which may be adjusted by the user (e.g., by clicking a restore down button or maximize button on the browser). The browser process 110 may also determine which portion of the web page is to be displayed within the browser window at a given time, which may be adjusted by the user (e.g., by scrolling the web page within the browser window). The browser process 110 may send this information to the render process 120 as view port information. As discussed further below, the render process 120 may use the view port information to determine where to display rendered web content on the display 160.

[0021] In one aspect, the render process 120 may initially receive HTML code for a web page from the browser process 110. The render process 120 may parse and interpret the received HTML code to determine how the various web content of the web page are to be rendered and

laid out for display to the user. For example, the render process 120 may utilize WebKit code to parse and interpret the HTML code or other markup language code.

[0022] If the HTML code includes one or more links to web content for the web page, then the render process 120 may send the one or more links to the browser process 110, and request that the browser process 110 retrieve the corresponding web content from the network 115. The browser process 110 may retrieve the corresponding web content by sending network requests to the network 115 using the one or more links. Upon receiving the requested web content from the network 115, the browser process 110 may send the web content to the render process 120 for rendering. If the HTML code includes a link to video content, then the render process 120 may instead send the link to the video process 140, which handles rendering of video content, as discussed further below.

[0023] The render process 120 may render some of the received web content (e.g., text and images) for the web page using software rendering. For example, the render process 120 may render some of the web content into one or more bitmaps, which may be stored in the shared memory. The render process 120 may also request that the video process 140 render other web content comprising video content based on a client-server model. In this example, the render process 120 acts as a client that requests services (e.g., video rendering) from the video process 140 and the video process 140 acts as a server that provides the requested services (e.g., video rendering), as discussed below.

[0024] In one aspect, the HTML code may include a link to video content, which may be displayed with other web content to the user (e.g., a web page with embedded video). In this regard, the HTML code may include a video tag for the video content. The video tag indicates that the web page includes video content and includes a video link (e.g., URL) to the source of the video content on the network 115. The render process 120 may determine that video content needs to be rendered based on the video tag and send the corresponding video link to the video process 140 for rendering of the video content. For each frame of video frame to be rendered, the render process 120 may also send the video process 140 a render target specifying a particular memory space in the GPU memory 155 in which to store the corresponding rendered video frame.

[0025] Upon receiving the video link from the render process 120, the video process 140 retrieves the video content from the network 115 using the video link. For example, the video process 140 may send a network request for the video content to the network 115 using the video link. After receiving the requested video content, the video process 140 may send the video content to the video hardware accelerator 145 for hardware-accelerated processing (e.g., video decoding), and instruct the hardware accelerator 145 to write the rendered video frame to the memory space in the GPU memory 155 specified by the render process 120. The video hardware accelerator 145 may store the rendered video frame as a graphics library (GL) texture in the GPU memory 155. The video hardware accelerator notifies the video process 140 when it is finished rendering the video frame, and the video process 140 notifies the render process 120 that the rendered video frame is ready for compositing.

[0026] The render process 120 may then instruct the GPU 150 to composite the web content rendered by the render process 120 and the video frame rendered by the video hardware accelerator 145 into a composite frame for display on the display 160. For the web content rendered by the render process 120, the render process 120 may send the GPU process 130 a pointer indicating where the rendered web content is stored in the shared memory. If the web content rendered by the render process 120 corresponds to two or more different layers, then the render process 120 may indicate where the rendered web content for each layer is stored in the shared memory. The render process 120 may also indicate an order in which the rendered web content for the different layers and the rendered video frame are to be composited (e.g., based on the HTML code, which may specify a hierarchy for the different layers of the web page).

[0027] The GPU process 130 may then use the pointer to locate the rendered web content in the shared memory and transfer the rendered web content to the GPU 150. The GPU processor 152 may convert the rendered web content into one or more GL textures and store the one or more GL textures in the GPU memory 155. If the rendered web content corresponds to two or more different layers, then the GPU process may instruct the GPU processor 152 to convert the rendered web content for each layer into a separate GL texture.

[0028] For the rendered video frame, the render process 120 may send the GPU process 130 the render target identifying the memory space in the GPU memory 155 holding the GL texture

for the rendered video frame. The GPU process 130 may then pass this information to the GPU processor 152 so that the GPU processor 152 can locate the GL texture for the rendered video frame in the GPU memory 155.

[0029] The render process 120 may also determine the layout of the web content in the web page (e.g., based on the corresponding HTML code) and where the web page is to be displayed on the display 160 (e.g., based on the view port information from the browser process 110), and send this information to the GPU process 130. Based on the received information, the GPU process 130 may issue commands (e.g., GL commands) to the GPU processor 152 specifying where to draw the GL textures for the rendered web content and the GL texture for the rendered video frame in the composite frame.

[0030] The GPU processor 152 may then build the composite frame in a display frame buffer. The display frame buffer may be a memory space allocated in the GPU memory 155 for the composite frame. To do this, the GPU processor 152 may write the GL textures for the rendered web content and the GL texture for the rendered video frame to the display frame buffer one at a time according to the commands from the GPU process 130. When the composite frame is completed in the display frame buffer, the composite frame may be outputted to the display 160 for display to the user.

[0031] In one aspect, the render process 120 may instruct the GPU process 130 with the order in which different layers of the web page are to be stacked in the composite frame (e.g., based on the HTML code, which may specify a hierarchy for the different layers of the web page). In this example, the rendered web content and the rendered video frame may correspond to different layers. According to the received instructions, the GPU process 150 may then send commands to the GPU processor 152 specifying the order in which to write the GL textures for the rendered web content and the GL texture for the rendered video frame to the display frame buffer. For example, the GPU processor 152 may write the GL textures in ascending order starting with the GL texture corresponding to the bottommost layer and ending with the GL texture corresponding to the uppermost layer. Two GL textures may overlap with each other. In this case, the GL texture corresponding to the upper layer is drawn on top of the GL texture corresponding to the bottom layer.

[0032] In one aspect, the render process 120 may also instruct the GPU process 150 where the rendered web content and rendered video frame are to be displayed on the display 160 (e.g., based on the view port information from the browser process 110). According to the received instructions, the GPU process 150 may send commands to the GPU 150 specifying where to write the GL textures for the rendered web content and the GL texture for the rendered video frame in the display frame buffer. The resulting composite frame may correspond to the portion of the display 160 in which the browser window is displayed. The composite frame may be further composited with other textures (e.g., texture corresponding to the browser tool bar, texture corresponding to a view of another application running on the computer system, etc.) by a compositor to form the entire frame displayed on the display 160 to the user.

[0033] The subject technology quickly and efficiently renders video content in a web page by creating a separate video process to handle rendering of the video content. The video process fetches the corresponding video content from the network and feeds the received video data to the video hardware accelerator 145, which provides fast and high performance video processing relative to a general-purpose processor. In addition, the video hardware accelerator 145 may write the rendered video data directly to the GPU memory 155, which is readily accessible by the GPU processor 152 for accelerated compositing of the rendered video data with the rest of the rendered web content into a composite frame for display to the user.

[0034] In one aspect, the render process 120 may be sandboxed, in which the render process 120 is restricted by a restriction policy mechanism from accessing certain system resources. For example, the render process 120 may be restricted from direct access to a network interface for communicating with the network 115. As a result, the render process 120 may be unable to directly retrieve content from the network 115 and may rely on another process (e.g., the browser process 110) to retrieve web content on its behalf. The render process 120 may also be restricted from directly accessing the GPU 150, directly accessing sensitive files, and/or other system resources. Sandboxing the render process 120 enhances security by preventing malicious code or un-trusted code in a web page from accessing the restricted resources.

[0035] In one aspect, the computing system 100 may include multiple render processes. For example, the computing system 100 may include a separate render process for each tab that is

open on the web browser. Each render process may be responsible for rendering a web page corresponding to the respective tab. The browser process 110 may trigger the creation of a new render process on the computing system 100 each time a new tab is opened. The browser process 110 may do this by instructing the processor (shown in FIG. 3) to load and execute code for implementing a render process. Thus, render processes may be created as needed during a browser session. An advantage of using multiple render processes is that, if one render process crashes, then the other render processes may not be affected. As a result, only the tab corresponding to the crashed render process may be terminated instead of all of the tabs.

[0036] In this aspect, multiple render processes may share the video process 140 for video rendering. The first render process requiring video rendering may trigger the creation of the video process 140 on the computing system 100 by instructing the processor to load and execute code for implementing the video process 140. In this aspect, each render process may act as a client of the video process 140, which provides video rendering services to the render processes.

[0037] In one aspect, the browser process 110 may send a cookie to the render process 120 indentifying the browser session used by the browser process 110 to request web content from the network 115. The render process 120 may then send the cookie to the video process 140 along with the video link to the video content. When the video process 140 requests the video content from the server hosting the video content, the video process 140 may also send the cookie to the server. The cookie allows the server to treat the request for the video content as from the same session requesting the web content, and to serve the video content properly.

[0038] FIG. 2 shows a method 200 for rendering content according to an aspect of the subject technology. The method 200 may be performed by the browser process 110, the render process 120, the video process 140 and the GPU process 130, all of which may be implemented on one or more processors.

[0039] In step 210, web content is retrieved from the network 115. For example, the browser process 110 may retrieve the web content by sending one or more network requests for web content to the network 115 and receiving the requested web content from the network 115.

[0040] In step 220, the retrieved web content is rendered into rendered web content. For example, the render process 120 may receive the retrieved web content from the browser process 110 and render the retrieved web content (e.g., using software-based renderer). The render process 120 may render the web content into a bitmap or other format, and store the rendered web content in the shared memory.

[0041] In step 230, video content is retrieved from the network 115. For example, the render process 120 may parse HTML code to retrieve a link (e.g., URL) to video content and send the retrieved link to the video process 140. The video process 140 may then retrieve the video content from the network 115 using the video link.

[0042] In step 240, the retrieved video content is directed to the video hardware accelerator 145 for video rendering. For example, the video process 140 may direct the retrieved video content to the video hardware accelerator 145, which performs video processing (e.g., video decoding) on the video content. The video hardware accelerator 145 may store the resulting rendered video frame in a memory space in the GPU memory 155.

[0043] In step 250, the GPU 150 is instructed to composite the rendered web content and the rendered video frame into a composite frame. For example, the GPU process 130 may transfer the rendered web content from the shared memory to the GPU 150 and instruct the GPU 150 to composite the transferred rendered web content and the rendered video frame stored in the GPU memory 155.

[0044] In one aspect, the video content may comprise a video data stream that is directed to the video hardware accelerator 145 by the video process 140 for rendering. The video hardware accelerator 145 may render the video data stream into a sequence of rendered video frames. For each rendered video frame, the video hardware accelerator 145 may notify the video process 140 when the rendered video frame is finished. The video process 140 may then notify the render process 120, which may then instruct the GPU process 130 to command the GPU 150 to composite the rendered video frame with the rest of the rendered web content. If the rendered web content does not change between video frames, then, for each new rendered video frame, the render process 120 may instruct the GPU process 130 to command the GPU 150 to composite the rendered web content already stored in the GPU 150 with the new rendered video frame.

[0045] In this aspect, for each rendered video frame, the video process 140 may instruct the video hardware accelerator 145 where to store the rendered video frame in the GPU memory 155. For example, two or more different memory spaces may be allocated in the GPU memory 155 for storing rendered video frames and each of the memory spaces may be identified by a unique identifier.

[0046] For each rendered video frame, the render process 130 may indicate to the video process 140 in which of the memory spaces the rendered video frame is to be stored using the corresponding identifier. The video process 140 may then communicate this information to the video hardware accelerator 145 so that the rendered video frame is stored in the appropriate memory space. When the rendered video frame is finished, the render process 140 may communicate the identity of the memory space to the GPU process 130 and instruct the GPU process 130 to command the GPU processor 152 to composite the rendered video frame in the identified memory space with the other rendered web content into a composite frame.

[0047] FIG. 3 conceptually illustrates an electronic system 300 with which some implementations of the subject technology are implemented. The electronic system 300 can be used to implement the computing system shown in FIG. 1. The electronic system 300 may be a smart phone, a tablet, a personal digital assistant, a laptop or other electronic system. While the electronic system 300 is shown in one configuration in FIG. 3, it is to be understood that the electronic system 300 may include additional, alternative and/or fewer components.

[0048] In the example shown in FIG. 3, the electronic system 300 includes a processor 310, system memory 315, a storage device 320, a network interface 330, an input interface 340, the video hardware accelerator 145, the GPU 150, the display 160, and a bus 370.

[0049] The bus 370 collectively represents all system, peripheral, and chipset buses that communicatively connect the numerous components of the electronic system 300. For instance, the bus 370 communicatively connects the processor 310 with the system memory 315 and the storage device 320. The processor 310 may retrieve instructions from one or more of these memories and execute the instructions to implement one or more of the processes according to various aspects of the subject technology. The processor 310 may comprise a single processor or a multi-core processor in different implementations.

[0050] The storage device 320 may comprise a solid state drive, a magnetic disk, or an optical drive. The storage device 320 may be used to store an operating system (OS), programs, and/or files. The system memory 315 may comprise volatile memory (e.g., a random access memory (RAM)) for storing instructions and data that the processor 310 needs at runtime.

[0051] In one aspect, the browser process 110, the render process 120, the graphics process unit (GPU) process 130, and the video process 140 may be implemented on the processor 310. For each process, the processor 310 may retrieve code corresponding to the process from a memory (e.g., system memory 315 and/or storage device 310) and execute the code to implement the process. The various processes running on the processor 300 may communicate with one another via inter-process communication (IPC). The shared memory used to share data across processes may be implemented using the system memory 315 and/or an internal memory of the processor 310.

[0052] The network interface 330 enables the processor 310 to communicate with the network 115 (e.g., a local area network (LAN), a wide area network (WAN), an intranet, the Internet, etc.). For example, the browser process 110 and the video process 140 implemented on the processor 310 may retrieve content from the network 115 via the network interface 330. The network interface 330 may include a wireless communication module for communicating with a base station or wireless access point 130 connected to the network 115 over a wireless link (WiFi wireless link, cellular wireless link, etc.).

[0053] The input interface 340 enables the user to communicate information and commands to the processor 310. For example, the input interface 340 may be coupled to an alphanumeric keyboard and/or a pointing device (e.g., touch pad or mouse) to receive commands from the user. For example, the browser process 110 may receive inputs from the user via the input interface 340. The browser process 110 may use the received inputs to determine when the user clicks on a link and to retrieve the corresponding web page from the network 115.

[0054] Many of the above-described features and applications may be implemented as a set of machine-readable instructions stored on a computer readable storage medium (also referred to as computer readable medium). When these instructions are executed by one or more processing unit(s) (e.g., one or more processors, cores of processors, or other processing units), they cause

the processing unit(s) to perform the actions indicated in the instructions. Examples of computer readable media include, but are not limited to, CD-ROMs, flash drives, RAM chips, hard drives, EPROMs, etc. The computer readable media does not include carrier waves and electronic signals passing wirelessly or over wired connections.

[0055] In this disclosure, the term "software" and "program" is meant to include firmware or applications stored in a memory, which can be executed by a processor. Also, in some implementations, multiple software aspects can be implemented as sub-parts of a larger program while remaining distinct software aspects. In some implementations, multiple software aspects can also be implemented as separate programs. Finally, any combination of separate programs that together implement a software aspect described here is within the scope of the disclosure. In some implementations, the software programs, when installed to operate on one or more electronic systems, define one or more specific machine implementations that execute and perform the operations of the software programs.

[0056] A computer program (also known as a program, software, software application, script, or code) can be written in any form of programming language, including compiled or interpreted languages, declarative or procedural languages, and it can be deployed in any form, including as a stand alone program or as a process, component, subroutine, object, or other unit suitable for use in a computing environment. A computer program may, but need not, correspond to a file in a file system. A program can be stored in a portion of a file that holds other programs or data (e.g., one or more scripts stored in a markup language document), in a single file dedicated to the program in question, or in multiple coordinated files (e.g., files that store one or more processes, sub programs, or portions of code). A computer program can be deployed to be executed on one computer or on multiple computers that are located at one site or distributed across multiple sites and interconnected by a communication network.

[0057] The functions described above can be implemented in digital electronic circuitry, in computer software, firmware or hardware. The techniques can be implemented using one or more computer program products. Programmable processors and computers can be included in or packaged as mobile devices. The processes and logic flows can be performed by one or more

programmable processors and by one or more programmable logic circuitry. General and special purpose computers and storage devices can be interconnected through communication networks.

[0058] Some implementations include electronic components, such as microprocessors, storage and memory that store computer program instructions in a machine-readable or computer-readable medium (alternatively referred to as computer-readable storage media, machine-readable media, or machine-readable storage media). Some examples of such computer-readable media include RAM, ROM, read-only compact discs (CD-ROM), recordable compact discs (CD-R), rewritable compact discs (CD-RW), read-only digital versatile discs (e.g., DVD-ROM, dual-layer DVD-ROM), a variety of recordable/rewritable DVDs (e.g., DVD-RAM, DVD-RW, DVD+RW, etc.), flash memory (e.g., SD cards, mini-SD cards, micro-SD cards, etc.), magnetic and/or solid state hard drives, read-only and recordable Blu-Ray® discs, ultra density optical discs, any other optical or magnetic media, and floppy disks. The computer-readable media can store a computer program that is executable by at least one processing unit and includes sets of instructions for performing various operations. Examples of computer programs or computer code include machine code, such as is produced by a compiler, and files including higher-level code that are executed by a computer, an electronic component, or a microprocessor using an interpreter.

[0059] While the above discussion primarily refers to microprocessor or multi-core processors that execute software, some implementations are performed by one or more integrated circuits, such as application specific integrated circuits (ASICs) or field programmable gate arrays (FPGAs). In some implementations, such integrated circuits execute instructions that are stored on the circuit itself.

[0060] As used in this specification and any claims of this application, the terms "computer", "processor", and "memory" all refer to electronic or other technological devices. These terms exclude people or groups of people. For the purposes of the specification, the terms display or displaying means displaying on an electronic device. As used in this specification and any claims of this application, the terms "computer readable medium" and "computer readable media" are entirely restricted to tangible, physical objects that store information in a form that is

readable by a computer. These terms exclude any wireless signals, wired download signals, and any other ephemeral signals.

[0061] It is understood that any specific order or hierarchy of steps in the processes disclosed is an illustration of exemplary approaches. Based upon design preferences, it is understood that the specific order or hierarchy of steps in the processes may be rearranged, or that all illustrated steps be performed. Some of the steps may be performed simultaneously. For example, in certain circumstances, multitasking and parallel processing may be advantageous. Moreover, the separation of various system components in the embodiments described above should not be understood as requiring such separation in all embodiments, and it should be understood that the described program components and systems can generally be integrated together in a single software product or packaged into multiple software products.

[0062] The previous description is provided to enable any person skilled in the art to practice the various aspects described herein. Various modifications to these aspects will be readily apparent to those skilled in the art, and the generic principles defined herein may be applied to other aspects. Thus, the claims are not intended to be limited to the aspects shown herein, but is to be accorded the full scope consistent with the language claims, wherein reference to an element in the singular is not intended to mean "one and only one" unless specifically so stated, but rather "one or more." Unless specifically stated otherwise, the term "some" refers to one or more.

[0063] A phrase such as an "aspect" does not imply that such aspect is essential to the subject technology or that such aspect applies to all configurations of the subject technology. A disclosure relating to an aspect may apply to all configurations, or one or more configurations. A phrase such as an aspect may refer to one or more aspects and vice versa. A phrase such as a "configuration" does not imply that such configuration is essential to the subject technology or that such configuration applies to all configurations of the subject technology. A disclosure relating to a configuration may apply to all configurations, or one or more configurations. A phrase such as a configuration may refer to one or more configurations and vice versa.

[0064] The word "exemplary" is used herein to mean "serving as an example or illustration." Any aspect or design described herein as "exemplary" is not necessarily to be construed as preferred or advantageous over other aspects or designs.

[0065] All structural and functional equivalents to the elements of the various aspects described throughout this disclosure that are known or later come to be known to those of ordinary skill in the art are expressly incorporated herein by reference and are intended to be encompassed by the claims. Moreover, nothing disclosed herein is intended to be dedicated to the public regardless of whether such disclosure is explicitly recited in the claims.

What is Claimed is:

1. A computer-implemented method for rendering a web page including web content and video content, the method comprising:

retrieving the web content from a network using a browser process implemented on one or more processors;

rendering the retrieved web content into rendered web content using a render process implemented on the one or more processors;

retrieving the video content from the network using a video process implemented on the one or more processors;

directing the retrieved video content to a video hardware accelerator, wherein the video hardware accelerator renders the video content into a rendered video frame; and

instructing a graphics processing unit (GPU) to composite the rendered web content and the rendered video frame into a composite frame for display to a user.

2. The method of claim 1, further comprising:

sending a render target to the video hardware accelerator using the video process, wherein the video hardware accelerator writes the rendered video frame to a memory space in the GPU specified by the render target; and

sending the render target to the GPU, wherein the GPU retrieves the rendered video frame from the memory space specified by the render target for compositing with the rendered web content.

3. The method of claim 2, further comprising:

generating the render target using the render process; and

sending the render target from the render process to the video process using inter-process communication (IPC).

4. The method of claim 1, wherein instructing the GPU to composite the rendered web content and the rendered video frame is performed using a GPU process implemented on the one or more processors.

5. The method of claim 4, further comprising:

storing the rendered web content in a shared memory using the render process, wherein the shared memory is shared by the render process and the GPU process; and

transferring the rendered web content from the shared memory to the GPU using the GPU process, wherein the GPU composites the rendered web content transferred from the shared memory with the rendered video frame.

6. The method of claim 1, further comprising:

parsing markup language code of the web page to retrieve a video link using the render process; and

sending the video link from the render process to the video process via inter-process communication, wherein the video process retrieves the video content from the network using the video link.

7. The method of claim 6, wherein the video link comprises a uniform resource locator (URL).

8. The method of claim 6, further comprising:

sending a cookie to a server on the network hosting the video content using the video process, wherein the cookie identifies a same browser session used by the browser process to retrieve the web content from the network.

9. The method of claim 6, further comprising:

parsing the markup language code of the web page to retrieve one or more web links using the render process; and

sending the one or more web links from the render process to the browser process via inter-process communication, wherein the browser process retrieves the web content from the network using the one or more web links.

10. The method of claim 9, wherein one or more web links comprise uniform resource locators (URLs).

11. The method of claim 1, wherein the video hardware accelerator performs video decoding on the video content to produce the rendered video frame.

12. A system for rendering a web page including web content and video content, the system comprising:

one or more processors;

a video hardware accelerator;

a graphics processing unit (GPU); and

a machine-readable medium comprising instructions stored therein, which when executed by the one or more processors, cause the one or more processors to perform operations comprising:

retrieving the web content from a network;

rendering the retrieved web content into rendered web content;

retrieving the video content from the network;

directing the retrieved video content to the video hardware accelerator, wherein the video hardware accelerator renders the video content into a rendered video frame and writes the rendered video frame to a memory space in the GPU; and

instructing the GPU to composite the rendered video frame in the memory space and the rendered web content into a composite frame for display to a user.

13. The system of claim 12, wherein the operations further comprise:

sending a render target to the video hardware accelerator, wherein the render target specifies the memory space to which the rendered video frame is written; and

sending the render target to the GPU, wherein the GPU retrieves the rendered video frame from the memory space specified by the render target for compositing with the rendered web content.

14. The system of claim 12, wherein the operations further comprise:

parsing markup language code of the web page to retrieve a video link, wherein the video link is used to retrieve the video content from the network.

15. The system of claim 14, wherein the video link comprises a uniform resource locator (URL).

16. The system of claim 14, the operations further comprise:

sending a cookie to a server on the network hosting the video content using the video process, wherein the cookie identifies a same browser session used by the browser process to retrieve the web content from the network.

17. The system of claim 12, wherein the video hardware accelerator performs video decoding on the video content to produce the rendered video frame.

18. The system of claim 12, wherein the operations further comprise:

writing the rendered web content to a memory accessible by the one or more processors; and

transferring the rendered web content from the memory to the GPU, wherein the GPU composites the rendered web content transferred from the memory with the rendered video frame.

19. The system of claim 12, wherein the operations further comprise:

processing markup language code of the web page to determine an order of the web content and the video content; and

instructing the GPU to composite the rendered web content and the rendered video frame based on the determined order.

20. A machine-readable medium comprising instructions stored therein, which when executed by a machine, cause the machine to perform operations for rendering a web page including web content and video content, the operations comprising:

retrieving the web content from a network;

rendering the retrieved web content into rendered web content;

storing the rendered web content in a memory;

retrieving the video content from the network;

directing the retrieved video content to a video hardware accelerator, wherein the video hardware accelerator renders the video content into a rendered video frame and writes the rendered video frame to a memory space in a graphics processing unit (GPU);

transferring the rendered web content from the memory to the GPU; and

instructing the GPU to composite the rendered web content transferred from the memory and the rendered video frame in the memory space in the GPU into a composite frame for display to a user.

21. The machine-readable medium of claim 20, wherein the operations further comprise:

sending a render target to the video hardware accelerator, wherein the render target specifies the memory space to which the rendered video frame is written; and

sending the render target to the GPU, wherein the GPU retrieves the rendered video frame from the memory space specified by the render target for compositing with the rendered web content.

22. The machine-readable medium of claim 20, wherein the operations further comprise:

parsing markup language code to retrieve a video link, wherein the video link is used to retrieve the video content from the network.

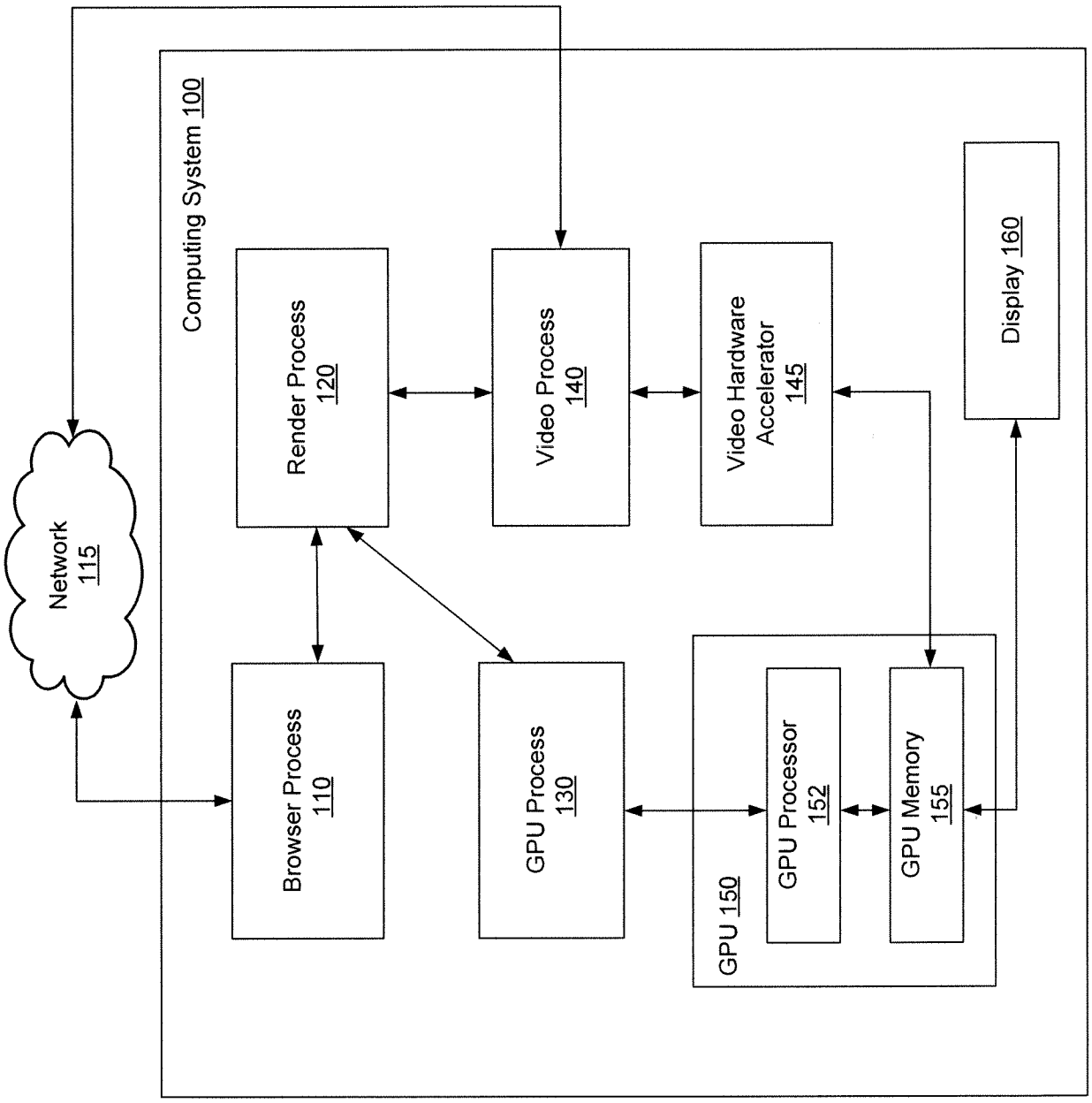
Computing System 100
Network 115
Render Process 120
Video Process 140
Video Hardware Accelerator 145
Display 160
Browser Process 110
GPU Process 130
GPU 150
GPU Processor 152
GPU Memory 155

FIG. 1

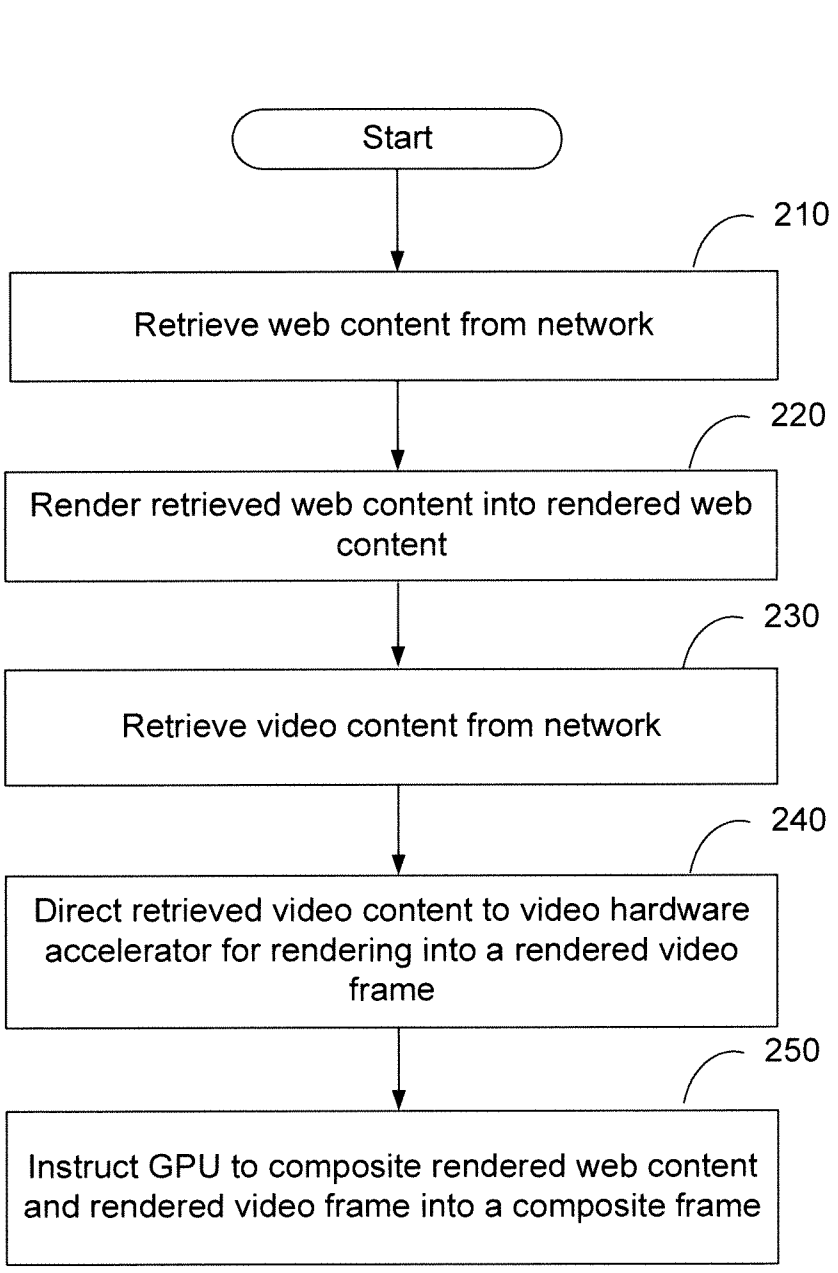
200
Start
210
Retrieve web content from network
220
Render retrieved web content into rendered web content
230
Retrieve video content from network
240
Direct retrieved video content to video hardware accelerator for rendering into a rendered video frame
250
Instruct GPU to composite rendered web content and rendered video frame into a composite frame

FIG. 2

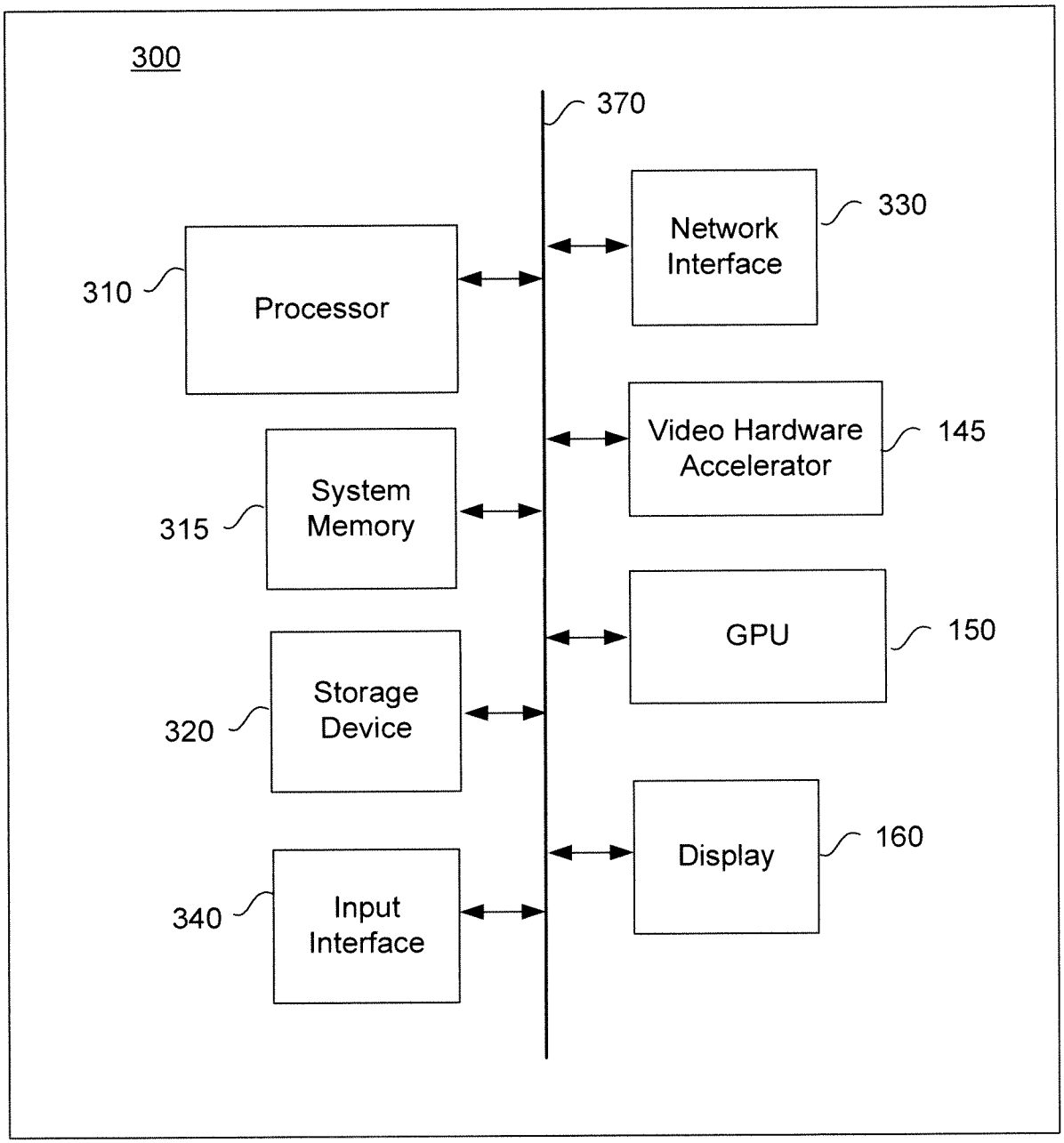
300
370
Network Interface
330
310
Processor
Video Hardware Accelerator
145
System Memory
315
GPU
150
Storage Device
320
Display
160
Input Interface
340

FIG. 3

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US13/22805

A. CLASSIFICATION OF SUBJECT MATTER
IPC(8) - G06F 3/00 (2013.01)
USPC - 715/760
According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC(8) Classification(s): G06F 3/00, 17/30, 15/16 (2013.01)
USPC Classification(s): 715/760, 234, 236, 277, 825, 843

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
MicroPatent (US-G, US-A, EP-A, EP-B, WO, JP-bib, DE-C,B, DE-A, DE-T, DE-U, GB-A, FR-A); IEEE/IEEEXplore; Google/Google Scholar; IP.com; Search Terms Used: render, create, website, web page, iGPU, data, hardware accelerator, content, video, video data, markup language, determine, order, sequence, inter-process communication, browser, cookie, session, identify, html, metadata,

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X -- Y	US 2010/0153692 A1 (KOTA B, et al.) June 17, 2010; abstract, paragraphs [0002], [0021], [0133], [0137], [0167], [0176]	1, 4, 5, 11, 12, 14, 17, 18, 20, 22 --- 2, 3, 6-10, 13, 15, 16, 19, 21
Y	US 8037401 B2 (LIPTON, DI) October 11, 2011; column 4, lines 20-24, column 5, lines 5-19,	2, 3, 6-10, 13, 15, 16, 21
Y	US 8032626 B1 (RUSSELL, EG et al.) October 4, 2011; column 12, lines 24-27, column 12, lines 32-41	8, 16
Y	US 2008/0120626 A1 (GRAFFAGNINO, P et al.) May 22, 2008; paragraph [0027]	19

☐ Further documents are listed in the continuation of Box C. ☐

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
16 March 2013 (16.03.2013)	02 APR 2013
Name and mailing address of the ISA/US Mail Stop PCT, Attn: ISA/US, Commissioner for Patents P.O. Box 1450, Alexandria, Virginia 22313-1450 Facsimile No. 571-273-3201	Authorized officer: Shane Thomas PCT Helpdesk: 571-272-4300 PCT OSP: 571-272-7774

Form PCT/ISA/210 (second sheet) (July 2009)