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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
07 January 2021 (07.01.2021)

WIPO | PCT

(10) International Publication Number
WO 2021/000220 A1

(51) International Patent Classification:
G06F 3/14 (2006.01)

(21) International Application Number:
PCT/CN2019/094190

(22) International Filing Date:
01 July 2019 (01.07.2019)

(25) Filing Language: English

(26) Publication Language: English

(71) Applicant: **QUALCOMM INCORPORATED** [US/US]; International Ip Administration 5775 Morehouse Drive, San Diego, Ca 92121-1714 (US).

(72) Inventors; and

(71) Applicants *(for US only)*: **ZHANG, Bin** [CN/CN]; International Ip Administration 5775 Morehouse Drive, San Diego, Ca 92121-1714 (US). **FANG, Sheng** [CN/CN]; International Ip Administration 5775 Morehouse Drive, San Diego, Ca 92121-1714 (US). **FU, Zhuo** [CN/CN]; International Ip Administration 5775 Morehouse Drive, San Diego, Ca 92121-1714 (US). **WANG, Jun** [CN/CN]; International Ip Administration 5775 Morehouse Drive, San Diego, Ca 92121-1714 (US).

(74) Agent: **LEE AND LI - LEAVEN IPR AGENCY LTD.**; Unit 2202, Tower A, Beijing Marriott Center, No. 7, Jian Guo Men South Avenue, Beijing 100005 (CN).

(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, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME,

(54) Title: METHODS AND APPARATUS FOR DYNAMIC JANK REDUCTION

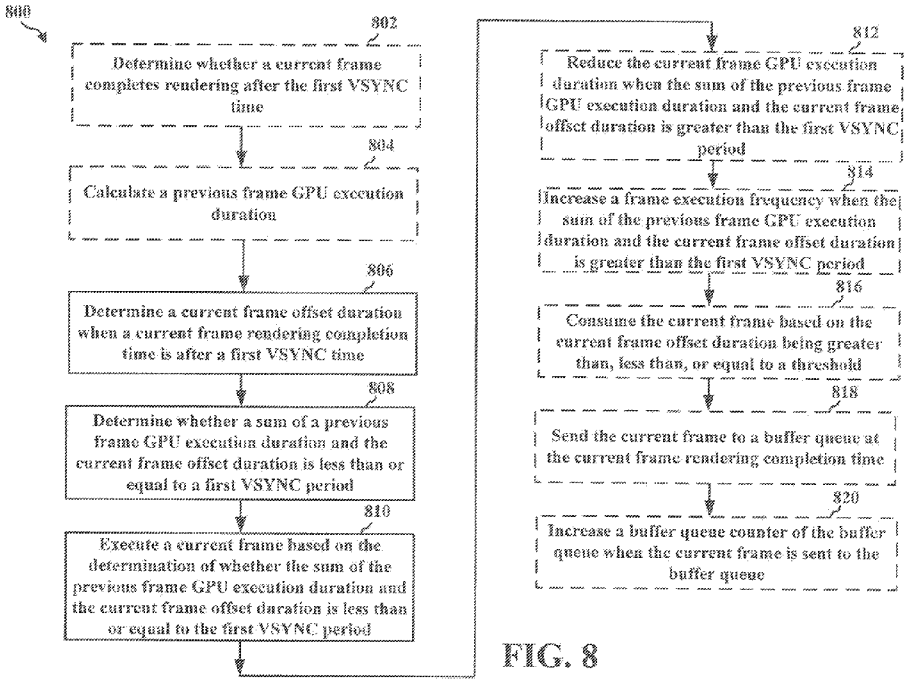

FIG. 8

(57) **Abstract:** The present disclosure relates to methods and apparatus for frame processing. The apparatus can determine a current frame offset duration when a current frame rendering completion time is after a first VSYNC time. In some aspects, the current frame offset duration can be equal to a difference between the first VSYNC time and the current frame rendering completion time. The apparatus can also determine whether a sum of a previous frame GPU execution duration and the current frame offset duration is less than or equal to a first VSYNC period. In some aspects, the first VSYNC period can begin at the first VSYNC time and ends at a second VSYNC time. Additionally, the apparatus can execute a current frame based on the determination of whether the sum of the previous frame GPU execution duration and the current frame offset duration is less than or equal to the first VSYNC period.

WO 2021/000220 A1

[Continued on next page]

MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, 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, ST, 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, KM, ML, MR, NE, SN, TD, TG).

Declarations under Rule 4.17:

— *of inventorship (Rule 4.17(iv))*

Published:

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

METHODS AND APPARATUS FOR DYNAMIC JANK REDUCTION

TECHNICAL FIELD

[0001] The present disclosure relates generally to processing systems and, more particularly, to one or more techniques for frame or graphics processing.

INTRODUCTION

[0002] Computing devices often utilize a graphics processing unit (GPU) to accelerate the rendering of graphical data for display. Such computing devices may include, for example, computer workstations, mobile phones such as so-called smartphones, embedded systems, personal computers, tablet computers, and video game consoles. GPUs execute a graphics processing pipeline that includes one or more processing stages that operate together to execute graphics processing commands and output a frame. A central processing unit (CPU) may control the operation of the GPU by issuing one or more graphics processing commands to the GPU. Modern day CPUs are typically capable of concurrently executing multiple applications, each of which may need to utilize the GPU during execution. A device that provides content for visual presentation on a display generally includes a GPU.

[0003] Typically, a GPU of a device is configured to perform the processes in a graphics processing pipeline. However, with the advent of wireless communication and smaller, handheld devices, there has developed an increased need for improved graphics processing.

SUMMARY

[0004] The following presents a simplified summary of one or more aspects in order to provide a basic understanding of such aspects. This summary is not an extensive overview of all contemplated aspects, and is intended to neither identify key elements of all aspects nor delineate the scope of any or all aspects. Its sole purpose is to present some concepts of one or more aspects in a simplified form as a prelude to the more detailed description that is presented later.

[0005] In an aspect of the disclosure, a method, a computer-readable medium, and an apparatus are provided. The apparatus may be a frame processor, a graphics processing unit (GPU), a frame composer, or a display processor. The apparatus can

determine a current frame offset duration when a current frame rendering completion time is after a first VSYNC time. In some aspects, the current frame offset duration can be equal to a difference between the first VSYNC time and the current frame rendering completion time. The apparatus can also determine whether a sum of a previous frame GPU execution duration and the current frame offset duration is less than or equal to a first VSYNC period. In some aspects, the first VSYNC period can begin at the first VSYNC time and ends at a second VSYNC time. Additionally, the apparatus can execute a current frame based on the determination of whether the sum of the previous frame GPU execution duration and the current frame offset duration is less than or equal to the first VSYNC period. The apparatus can also reduce the current frame GPU execution duration when the sum of the previous frame GPU execution duration and the current frame offset duration is greater than the first VSYNC period. Further, the apparatus can increase a frame execution frequency when the sum of the previous frame GPU execution duration and the current frame offset duration is greater than the first VSYNC period. The apparatus can also determine whether a current frame completes rendering after the first VSYNC time. Also, the apparatus can calculate the previous frame GPU execution duration.

[0006] The details of one or more examples of the disclosure are set forth in the accompanying drawings and the description below. Other features, objects, and advantages of the disclosure will be apparent from the description and drawings, and from the claims.

BRIEF DESCRIPTION OF DRAWINGS

[0007] FIG. 1 is a block diagram that illustrates an example content generation system in accordance with one or more techniques of this disclosure.

[0008] FIG. 2 illustrates an example GPU in accordance with one or more techniques of this disclosure.

[0009] FIG. 3 illustrates an example timing diagram in accordance with one or more techniques of this disclosure.

[0010] FIG. 4 illustrates an example timing diagram in accordance with one or more techniques of this disclosure.

[0011] FIG. 5 illustrates an example timing diagram in accordance with one or more techniques of this disclosure.

[0012] FIG. 6 illustrates an example timing diagram in accordance with one or more techniques of this disclosure.

[0013] FIG. 7 illustrates an example timing diagram in accordance with one or more techniques of this disclosure.

[0014] FIG. 8 illustrates an example flowchart of an example method in accordance with one or more techniques of this disclosure.

DETAILED DESCRIPTION

[0015] Some aspects of janks reduction technology may be dependent upon certain platforms or scenarios. This may result in the janks reduction technology being less accurate and/or use an unnecessary amount of power. However, aspects of the present disclosure may make the threshold for triggering janks reduction technology platform independent. Further, aspects of the present disclosure may result in janks reduction mechanisms that can be changed dynamically and/or automatically. In turn, this can allow janks detection mechanisms to become more accurate and/or less dependent on certain scenarios. Aspects of the present disclosure can utilize jank reduction technology that is based on a GPU execution or latency prediction. For instance, aspects of the present disclosure may utilize the execution or latency duration of a previous frame in order to predict the execution or latency duration for a current frame. For example, as the content for two consecutive frames may be similar, this can result in similar latencies between the frames. By utilizing execution or latency times for previous frames, the jank reduction technology herein can be more accurate and/or utilize less power.

[0016] Various aspects of systems, apparatuses, computer program products, and methods are described more fully hereinafter with reference to the accompanying drawings. This disclosure may, however, be embodied in many different forms and should not be construed as limited to any specific structure or function presented throughout this disclosure. Rather, these aspects are provided so that this disclosure will be thorough and complete, and will fully convey the scope of this disclosure to those skilled in the art. Based on the teachings herein one skilled in the art should appreciate that the scope of this disclosure is intended to cover any aspect of the systems, apparatuses, computer program products, and methods disclosed herein, whether implemented independently of, or combined with, other aspects of the disclosure. For example, an apparatus may be implemented or a method may be practiced using any number of

the aspects set forth herein. In addition, the scope of the disclosure is intended to cover such an apparatus or method which is practiced using other structure, functionality, or structure and functionality in addition to or other than the various aspects of the disclosure set forth herein. Any aspect disclosed herein may be embodied by one or more elements of a claim.

[0017] Although various aspects are described herein, many variations and permutations of these aspects fall within the scope of this disclosure. Although some potential benefits and advantages of aspects of this disclosure are mentioned, the scope of this disclosure is not intended to be limited to particular benefits, uses, or objectives. Rather, aspects of this disclosure are intended to be broadly applicable to different wireless technologies, system configurations, networks, and transmission protocols, some of which are illustrated by way of example in the figures and in the following description. The detailed description and drawings are merely illustrative of this disclosure rather than limiting, the scope of this disclosure being defined by the appended claims and equivalents thereof.

[0018] Several aspects are presented with reference to various apparatus and methods. These apparatus and methods are described in the following detailed description and illustrated in the accompanying drawings by various blocks, components, circuits, processes, algorithms, and the like (collectively referred to as "elements"). These elements may be implemented using electronic hardware, computer software, or any combination thereof. Whether such elements are implemented as hardware or software depends upon the particular application and design constraints imposed on the overall system.

[0019] By way of example, an element, or any portion of an element, or any combination of elements may be implemented as a "processing system" that includes one or more processors (which may also be referred to as processing units). Examples of processors include microprocessors, microcontrollers, graphics processing units (GPUs), general purpose GPUs (GPGPUs), central processing units (CPUs), application processors, digital signal processors (DSPs), reduced instruction set computing (RISC) processors, systems-on-chip (SOC), baseband processors, application specific integrated circuits (ASICs), field programmable gate arrays (FPGAs), programmable logic devices (PLDs), state machines, gated logic, discrete hardware circuits, and other suitable hardware configured to perform the various functionality described throughout this disclosure. One or more processors in the

processing system may execute software. Software can be construed broadly to mean instructions, instruction sets, code, code segments, program code, programs, subprograms, software components, applications, software applications, software packages, routines, subroutines, objects, executables, threads of execution, procedures, functions, etc., whether referred to as software, firmware, middleware, microcode, hardware description language, or otherwise. The term application may refer to software. As described herein, one or more techniques may refer to an application, i.e., software, being configured to perform one or more functions. In such examples, the application may be stored on a memory, e.g., on-chip memory of a processor, system memory, or any other memory. Hardware described herein, such as a processor may be configured to execute the application. For example, the application may be described as including code that, when executed by the hardware, causes the hardware to perform one or more techniques described herein. As an example, the hardware may access the code from a memory and execute the code accessed from the memory to perform one or more techniques described herein. In some examples, components are identified in this disclosure. In such examples, the components may be hardware, software, or a combination thereof. The components may be separate components or sub-components of a single component.

[0020] Accordingly, in one or more examples described herein, the functions described may be implemented in hardware, software, or any combination thereof. If implemented in software, the functions may be stored on or encoded as one or more instructions or code on a computer-readable medium. Computer-readable media includes computer storage media. Storage media may be any available media that can be accessed by a computer. By way of example, and not limitation, such computer-readable media can comprise a random access memory (RAM), a read-only memory (ROM), an electrically erasable programmable ROM (EEPROM), optical disk storage, magnetic disk storage, other magnetic storage devices, combinations of the aforementioned types of computer-readable media, or any other medium that can be used to store computer executable code in the form of instructions or data structures that can be accessed by a computer.

[0021] In general, this disclosure describes techniques for having a graphics processing pipeline in a single device or multiple devices, improving the rendering of graphical content, and/or reducing the load of a processing unit, i.e., any processing unit configured to perform one or more techniques described herein, such as a GPU. For

example, this disclosure describes techniques for graphics processing in any device that utilizes graphics processing. Other example benefits are described throughout this disclosure.

[0022] As used herein, instances of the term "content" may refer to "graphical content," "image," and vice versa. This is true regardless of whether the terms are being used as an adjective, noun, or other parts of speech. In some examples, as used herein, the term "graphical content" may refer to a content produced by one or more processes of a graphics processing pipeline. In some examples, as used herein, the term "graphical content" may refer to a content produced by a processing unit configured to perform graphics processing. In some examples, as used herein, the term "graphical content" may refer to a content produced by a graphics processing unit.

[0023] In some examples, as used herein, the term "display content" may refer to content generated by a processing unit configured to perform displaying processing. In some examples, as used herein, the term "display content" may refer to content generated by a display processing unit. Graphical content may be processed to become display content. For example, a graphics processing unit may output graphical content, such as a frame, to a buffer (which may be referred to as a framebuffer). A display processing unit may read the graphical content, such as one or more frames from the buffer, and perform one or more display processing techniques thereon to generate display content. For example, a display processing unit may be configured to perform composition on one or more rendered layers to generate a frame. As another example, a display processing unit may be configured to compose, blend, or otherwise combine two or more layers together into a single frame. A display processing unit may be configured to perform scaling, e.g., upscaling or downscaling, on a frame. In some examples, a frame may refer to a layer. In other examples, a frame may refer to two or more layers that have already been blended together to form the frame, i.e., the frame includes two or more layers, and the frame that includes two or more layers may subsequently be blended.

[0024] FIG. 1 is a block diagram that illustrates an example content generation system 100 configured to implement one or more techniques of this disclosure. The content generation system 100 includes a device 104. The device 104 may include one or more components or circuits for performing various functions described herein. In some examples, one or more components of the device 104 may be components of an SOC. The device 104 may include one or more components configured to perform

one or more techniques of this disclosure. In the example shown, the device 104 may include a processing unit 120, and a system memory 124. In some aspects, the device 104 can include a number of optional components, e.g., a communication interface 126, a transceiver 132, a receiver 128, a transmitter 130, a display processor 127, and one or more displays 131. Reference to the display 131 may refer to the one or more displays 131. For example, the display 131 may include a single display or multiple displays. The display 131 may include a first display and a second display. The first display may be a left-eye display and the second display may be a right-eye display. In some examples, the first and second display may receive different frames for presentment thereon. In other examples, the first and second display may receive the same frames for presentment thereon. In further examples, the results of the graphics processing may not be displayed on the device, e.g., the first and second display may not receive any frames for presentment thereon. Instead, the frames or graphics processing results may be transferred to another device. In some aspects, this can be referred to as split-rendering.

[0025] The processing unit 120 may include an internal memory 121. The processing unit 120 may be configured to perform graphics processing, such as in a graphics processing pipeline 107. In some examples, the device 104 may include a display processor, such as the display processor 127, to perform one or more display processing techniques on one or more frames generated by the processing unit 120 before presentment by the one or more displays 131. The display processor 127 may be configured to perform display processing. For example, the display processor 127 may be configured to perform one or more display processing techniques on one or more frames generated by the processing unit 120. The one or more displays 131 may be configured to display or otherwise present frames processed by the display processor 127. In some examples, the one or more displays 131 may include one or more of: a liquid crystal display (LCD), a plasma display, an organic light emitting diode (OLED) display, a projection display device, an augmented reality display device, a virtual reality display device, a head-mounted display, or any other type of display device.

[0026] Memory external to the processing unit 120, such as system memory 124, may be accessible to the processing unit 120. For example, the processing unit 120 may be configured to read from and/or write to external memory, such as the system memory 124. The processing unit 120 may be communicatively coupled to the system memory

124 over a bus. In some examples, the processing unit 120 may be communicatively coupled to each other over the bus or a different connection.

[0027] The internal memory 121 or the system memory 124 may include one or more volatile or non-volatile memories or storage devices. In some examples, internal memory 121 or the system memory 124 may include RAM, SRAM, DRAM, erasable programmable ROM (EPROM), electrically erasable programmable ROM (EEPROM), flash memory, a magnetic data media or an optical storage media, or any other type of memory.

[0028] The internal memory 121 or the system memory 124 may be a non-transitory storage medium according to some examples. The term "non-transitory" may indicate that the storage medium is not embodied in a carrier wave or a propagated signal. However, the term "non-transitory" should not be interpreted to mean that internal memory 121 or the system memory 124 is non-movable or that its contents are static. As one example, the system memory 124 may be removed from the device 104 and moved to another device. As another example, the system memory 124 may not be removable from the device 104.

[0029] The processing unit 120 may be a central processing unit (CPU), a graphics processing unit (GPU), a general purpose GPU (GPGPU), or any other processing unit that may be configured to perform graphics processing. In some examples, the processing unit 120 may be integrated into a motherboard of the device 104. In some examples, the processing unit 120 may be present on a graphics card that is installed in a port in a motherboard of the device 104, or may be otherwise incorporated within a peripheral device configured to interoperate with the device 104. The processing unit 120 may include one or more processors, such as one or more microprocessors, GPUs, application specific integrated circuits (ASICs), field programmable gate arrays (FPGAs), arithmetic logic units (ALUs), digital signal processors (DSPs), discrete logic, software, hardware, firmware, other equivalent integrated or discrete logic circuitry, or any combinations thereof. If the techniques are implemented partially in software, the processing unit 120 may store instructions for the software in a suitable, non-transitory computer-readable storage medium, e.g., internal memory 121, and may execute the instructions in hardware using one or more processors to perform the techniques of this disclosure. Any of the foregoing, including hardware, software, a combination of hardware and software, etc., may be considered to be one or more processors.

[0030] In some aspects, the content generation system 100 can include an optional communication interface 126. The communication interface 126 may include a receiver 128 and a transmitter 130. The receiver 128 may be configured to perform any receiving function described herein with respect to the device 104. Additionally, the receiver 128 may be configured to receive information, e.g., eye or head position information, rendering commands, or location information, from another device. The transmitter 130 may be configured to perform any transmitting function described herein with respect to the device 104. For example, the transmitter 130 may be configured to transmit information to another device, which may include a request for content. The receiver 128 and the transmitter 130 may be combined into a transceiver 132. In such examples, the transceiver 132 may be configured to perform any receiving function and/or transmitting function described herein with respect to the device 104.

[0031] Referring again to FIG. 1, in certain aspects, the graphics processing pipeline 107 may include a determination component 198 configured to determine a current frame offset duration when a current frame rendering completion time is after a first VSYNC time. In some aspects, the current frame offset duration can be equal to a difference between the first VSYNC time and the current frame rendering completion time. The determination component 198 can also be configured to determine whether a sum of a previous frame GPU execution duration and the current frame offset duration is less than or equal to a first VSYNC period. In some aspects, the first VSYNC period can begin at the first VSYNC time and ends at a second VSYNC time. Additionally, determination component 198 can be configured to execute a current frame based on the determination of whether the sum of the previous frame GPU execution duration and the current frame offset duration is less than or equal to the first VSYNC period. The determination component 198 can also be configured to reduce the current frame GPU execution duration when the sum of the previous frame GPU execution duration and the current frame offset duration is greater than the first VSYNC period. Further, the determination component 198 can be configured to increase a frame execution frequency when the sum of the previous frame GPU execution duration and the current frame offset duration is greater than the first VSYNC period. The determination component 198 can also be configured to determine whether a current frame completes rendering after the first VSYNC time. The determination component 198 can also be configured to calculate the previous frame GPU execution duration.

[0032] As described herein, a device, such as the device 104, may refer to any device, apparatus, or system configured to perform one or more techniques described herein. For example, a device may be a server, a base station, user equipment, a client device, a station, an access point, a computer, e.g., a personal computer, a desktop computer, a laptop computer, a tablet computer, a computer workstation, or a mainframe computer, an end product, an apparatus, a phone, a smart phone, a server, a video game platform or console, a handheld device, e.g., a portable video game device or a personal digital assistant (PDA), a wearable computing device, e.g., a smart watch, an augmented reality device, or a virtual reality device, a non-wearable device, a display or display device, a television, a television set-top box, an intermediate network device, a digital media player, a video streaming device, a content streaming device, an in-car computer, any mobile device, any device configured to generate graphical content, or any device configured to perform one or more techniques described herein. Processes herein may be described as performed by a particular component (e.g., a GPU), but, in further embodiments, can be performed using other components (e.g., a CPU), consistent with disclosed embodiments.

[0033] GPUs can process multiple types of data or data packets in a GPU pipeline. For instance, in some aspects, a GPU can process two types of data or data packets, e.g., context register packets and draw call data. A context register packet can be a set of global state information, e.g., information regarding a global register, shading program, or constant data, which can regulate how a graphics context will be processed. For example, context register packets can include information regarding a color format. In some aspects of context register packets, there can be a bit that indicates which workload belongs to a context register. Also, there can be multiple functions or programming running at the same time and/or in parallel. For example, functions or programming can describe a certain operation, e.g., the color mode or color format. Accordingly, a context register can define multiple states of a GPU.

[0034] Context states can be utilized to determine how an individual processing unit functions, e.g., a vertex fetcher (VFD), a vertex shader (VS), a shader processor, or a geometry processor, and/or in what mode the processing unit functions. In order to do so, GPUs can use context registers and programming data. In some aspects, a GPU can generate a workload, e.g., a vertex or pixel workload, in the pipeline based on the context register definition of a mode or state. Certain processing units, e.g., a VFD, can use these states to determine certain functions, e.g., how a vertex is assembled.

As these modes or states can change, GPUs may need to change the corresponding context. Additionally, the workload that corresponds to the mode or state may follow the changing mode or state.

[0035] FIG. 2 illustrates an example GPU 200 in accordance with one or more techniques of this disclosure. As shown in FIG. 2, GPU 200 includes command processor (CP) 210, draw call packets 212, VFD 220, VS 222, vertex cache (VPC) 224, triangle setup engine (TSE) 226, rasterizer (RAS) 228, Z process engine (ZPE) 230, pixel interpolator (PI) 232, fragment shader (FS) 234, render backend (RB) 236, L2 cache (UCHE) 238, and system memory 240. Although FIG. 2 displays that GPU 200 includes processing units 220-238, GPU 200 can include a number of additional processing units. Additionally, processing units 220-238 are merely an example and any combination or order of processing units can be used by GPUs according to the present disclosure. GPU 200 also includes command buffer 250, context register packets 260, and context states 261.

[0036] As shown in FIG. 2, a GPU can utilize a CP, e.g., CP 210, or hardware accelerator to parse a command buffer into context register packets, e.g., context register packets 260, and/or draw call data packets, e.g., draw call packets 212. The CP 210 can then send the context register packets 260 or draw call data packets 212 through separate paths to the processing units or blocks in the GPU. Further, the command buffer 250 can alternate different states of context registers and draw calls. For example, a command buffer can be structured in the following manner: context register of context N, draw call(s) of context N, context register of context N+1, and draw call(s) of context N+1.

[0037] The mobile gaming market is one of the most important markets in the mobile world. In this market, users care greatly about the game performance. A variety of factors can be performance indicators for the mobile gaming market. For instance, frames per second (FPS) and janks, i.e., delays or pauses in frame rendering or composition, are important key performance indicators (KPI) in this market. In some aspects, a jank can be a perceptible pause in the rendering of a software application's user interface. Both FPS and janks are KPIs in game performance and/or device display capability. In mobile gaming applications, janks can be the result of a number of factors, such as slow operations or poor interface design. In some instances, a jank can also correspond to a change in the refresh rate of the display at the device. Janks are important to mobile gaming because if the display fresh latency is not stable, this

can impact the user experience. Accordingly, some aspects of the mobile gaming industry are focused on reducing janks and increasing FPS.

[0038] In the mobile gaming industry, games can be run at a variety of different FPS modes. In some aspects, games can run at 30 FPS mode. In other aspects, games can run at different FPS modes, e.g., 20 or 60 FPS. Aspects of the present disclosure can include a current frame latency time, which can refer to the time difference between when a previous frame completes rendering a current frame completes rendering. The frame latency time can also refer to the time between successive refreshing frames. The frame latency time can also be based on a frame rate. In some aspects, when a gaming application runs at 30 FPS mode, although the average frame rate may be around 30 FPS, the frame latency time may not be stable. For instance, the frame latency time for each frame can be 33.33 ms (e.g., corresponding to 30 FPS), 16.67 ms (e.g., corresponding to 60 FPS), or 50 ms (e.g., corresponding to 20 FPS).

[0039] Jank reduction technology can be utilized in a number of different scenarios. For instance, slow frames, e.g., frames under 30 FPS, may optimize janks reduction differently than fast frames. For example, there may be frame pacing issues for frames under 30 FPS, which may utilize a different janks reduction technology than faster frames. In some aspects, different mechanisms or designs may have the ability to detect janks. Also, once janks are detected, other mechanisms can be triggered, e.g., a frame composer or surface flinger (SF) mechanism can be directly triggered to bypass a vertical synchronization (VSYNC) time in order to avoid janks. In some aspects, the threshold of the janks reduction technology may be platform dependent, which may need certain tuning efforts.

[0040] Aspects of the present disclosure may make janks reduction thresholds platform independent, such that they can be changed dynamically and/or automatically. In some instances, aspects of the present disclosure may attempt to reduce janks in a number of different application, e.g., games and other applications. For example, a frame composer or smart surface flinger (SSF) may bypass a VSYNC time under specific scenarios, e.g., in 60 FPS mode. Additionally, if a frame is not ready to be consumed at a certain VSYNC time, e.g., it is within a threshold, then the frame composer or SF mechanism may be triggered directly in order to reduce janks.

[0041] In some aspects, the threshold for triggering janks reduction technology may be dependent upon certain platforms. This may result in the janks reduction technology being less accurate and/or more dependent on certain platforms or scenarios.

However, aspects of the present disclosure may make the threshold for triggering janks reduction technology platform independent. Accordingly, aspects of the present disclosure may result in a more universally adaptable janks reduction technology. Further, aspects of the present disclosure may result in janks reduction mechanisms that can be changed dynamically and/or automatically. In turn, this can allow janks detection mechanisms to become more accurate and/or less dependent on certain platforms or scenarios. For example, a threshold that triggers a frame composer or SF mechanism may be independent. By doing so, such optimization can be applicable to a number of different FPS modes, e.g., 30, 60, 90, and/or 120 FPS mode.

[0042] As mentioned herein, janks can be caused by a number of different issues, such as GPU latency in different FPS modes, e.g., in 30 or 60 FPS mode. For instance, graphics dynamic clock and voltage scaling (DCVS) may not have input from FPS information or a VSYNC timestamp. For example, if a main input is a GPU workload, then in certain game scenarios, the GPU may miss a tearing effect (TE) or display a VSYNC frequently. Some aspects of the present disclosure may utilize jank reduction technology focused on fixing such issues. In some aspects, a frame composer or SF mechanism can be refreshed at a consistent time frame, e.g., every 30 FPS. However, a display pipeline may experience a jank even with a consistent frame composer or SF mechanism. For example, a GPU may miss a hardware VSYNC signal, which can cause a jank.

[0043] As indicated herein, if a frame takes too long to be rendered and is not ready for transmission to a display at a scheduled VSYNC time, this can result in a delayed frame display time and a corresponding jank. As such, janks can be the result of a delayed frame rendering. In some aspects, a frame buffer or buffer queue can queue frames waiting to be sent to the display. If a frame takes too long to be rendered, then the frame may not be consumed or sent to the buffer queue by the scheduled VSYNC time. In some aspects, a frame composer or SF mechanism can consume the frame or help send the frame to the buffer queue or display. If the renderer takes too long to render a frame, then the frame composer or SF mechanism may be delayed in consuming the frame, so the frame will be delayed in being transmitted to the display. As such, a delay in rendering can cause a resulting delay in frame consumption or display transmission. In some aspects, if a frame has not finished rendering by a scheduled VSYNC time, then the frame will not be consumed by the composer until the next VSYNC time. In these aspects, if there are no frames in the buffer queue,

then the frame composer or SF mechanism may not be triggered to consume the frame. As the frame is not consumed, this can result in a jank.

[0044] FIG. 3 illustrates an example timing diagram 300 in accordance with one or more techniques of this disclosure. As shown in FIG. 3, timing diagram 300 includes first frame 301, second frame 302, third frame 303, fourth frame 304, fifth frame 305, sixth frame 306, VSYNC time period (VSYNC_period) 310, first VSYNC time 321, second VSYNC time 322, third VSYNC time 323, fourth VSYNC time 324, fifth VSYNC time 325, sixth VSYNC time 326, seventh VSYNC time 327, and eighth VSYNC time 328. FIG. 3 also displays a renderer, a user interface (UI), a GPU, a frame composer or SF mechanism, a display or display engine, and a VSYNC timing mechanism. The display or display engine can also be referred to as other terms, such as a display buffer.

[0045] FIG. 3 displays jank reduction technology in certain FPS modes, e.g., 60 FPS mode, of a mobile gaming application. As shown in FIG. 3, VSYNC time period 310 can correspond to the same amount of time between successive VSYNC times. In some aspects, VSYNC time period 310 can be 16.67 ms. As shown in FIG. 3, when the buffer is ready, the frame composer or SF mechanism can be triggered directly. Additionally, a jank can be eliminated if a fence of a kernel graphics support layer (KGSL) is signaled before a TE arrives. Further, the SF mechanism can consume a frame if the frame, e.g., frame 303, is rendered within a certain threshold.

[0046] As mentioned above, FIG. 3 displays jank reduction technology in certain FPS modes, e.g., 60 FPS mode. In some aspects, if a frame takes a long time to render and results in a missed VSYNC time, a potential jank can still be eliminated. For instance, eliminating a potential jank may depend on whether a rendering task is completed within a certain threshold. This threshold may be hardcoded and include tuning efforts on specific platforms. Also, a potential jank may be eliminated if a GPU can finish executing a frame before a TE arrives. In some aspects, the present disclosure may not be able to determine if the GPU execution for a current frame can finish within a certain time frame.

[0047] As shown in FIG. 3, the renderer may take a long time to render frame 303, so the frame 303 may not complete rendering prior to the fourth VSYNC time 324. Based on this, the SF mechanism can be triggered directly, which may then cause frame 303 to be sent to the display. Accordingly, a potential jank can be eliminated if the SF mechanism is triggered within a threshold time. In timing diagram 300, this threshold

may be hard coded, and may not be triggered dynamically. As mentioned above, other aspects of jank reduction technology may utilize a dynamic threshold to replace a hardcoded threshold, which can result in a number of advantages and benefits, such as jank reduction technology that is more accurate and power friendly.

[0048] Aspects of the present disclosure can utilize jank reduction technology that is based on a GPU latency prediction. For instance, a GPU latency prediction can show a certain redundancy between consecutive or neighboring frames. In some instances, the GPU latency between two consecutive frames can be similar. Indeed, as the frame content can be similar between consecutive frames, then the frame latency may also be similar. For example, the latency for neighboring frames can be 9.65 ms and 9.54 ms, respectively. As the content for two neighboring frames may be similar, merely a few small objects may need updating between frames, which can result in similar latencies between the frames. Additionally, from the work load perspective, both the CPU and GPU load may be similar for neighboring frames. As latencies can be similar for consecutive frames, aspects of the present disclosure may predict the latency for a subsequent frame based on the latency of the previous frame.

[0049] In some aspects, in order to determine if consecutive frames are similar, the present disclosure may determine the draw call status or the number of each frame. As such, aspects of the present disclosure may determine the draw call information for each frame in order to determine if a neighboring frame is similar. For instance, there may be a correlation between the amount of draw calls and the GPU execution latency and/or frequency. For example, aspects of the present disclosure can determine the draw call number of each frame and take a screenshot after a certain amount of time, e.g., every second. By doing so, aspects of the present disclosure can determine the draw call information for two neighboring frames. When the draw call information for consecutive frames is similar, e.g., the biggest gap between consecutive frames may be about 4.5%, the draw call difference between the frames may be negligible. This also manifests that the GPU workload for consecutive frames can be similar. Accordingly, the GPU execution time or latency for neighboring frames may be similar.

[0050] As indicated herein, aspects of the present disclosure can use the latency of a previous frame to predict the latency of a subsequent or current frame. Aspects of the present disclosure can also utilize the GPU workload of a previous frame to predict the GPU workload of a current frame. Thus, the present disclosure can use information from a

previous frame to predict the GPU workload and latency for the current frame. With this information, aspects of the present disclosure can dynamically change a certain threshold, which can make the jank detection process more accurate and/or power friendly. Accordingly, aspects of the present disclosure can utilize information from previous frames in order to predict the current frame latency and/or dynamically change certain thresholds.

[0051] FIG. 4 illustrates an example timing diagram 400 in accordance with one or more techniques of this disclosure. As shown in FIG. 4, timing diagram 400 includes first frame 401, second frame 402, VSYNC time period (VSYNC_period) 410, first VSYNC time 421, second VSYNC time 422, third VSYNC time 423, fourth VSYNC time 424, and fifth VSYNC time 425. FIG. 4 also displays a GPU, a frame composer or SF mechanism, a SF binder, a display or display engine, and a buffer queue. The display or display engine can also be referred to as other terms, such as a display buffer.

[0052] As shown in FIG. 4, aspects of the present disclosure can predict the GPU execution or latency duration for a current frame, e.g., frame 402, based on the GPU execution or latency duration of a previous frame, e.g., frame 401. Indeed, aspects of the present disclosure can use the GPU execution or latency times of some frames to predict these times for consecutive frames. So the present disclosure can predict the current frame latency based on the previous frame latency. As indicated in FIG. 4, the predicted latency or execution duration for frame 402 can be equal to the latency or execution duration for frame 401. The latency or execution duration can also be referred to as a GPU latency or execution duration. In some aspects, timing diagram 400 can illustrate jank reduction technology based on GPU latency prediction for certain FPS modes, e.g., 30 FPS mode. FIG. 4 also shows a number of different terms: T_{vsync} is the timestamp for a previous VSYNC time, e.g., VSYNC time 422 for frame 402, T_{qb} is the timestamp when frame 402 is ready, e.g., when SF binder has finished frame 402, T_{qb}-T_{vsync} is the latency compared with the previous VSYNC signal, e.g., VSYNC time 422 for frame 402, $T_{gpu\text{-}predict}$ is the predicted latency of a current frame, e.g., predicted latency for frame 402, and $T_{GPU\text{-}Frame1}$ is the latency of a previous frame, e.g., the latency of frame 401.

[0053] FIG. 4 also displays a number of different formulas or calculations. For instance, $T_{gpu\text{-}predict}$ is equal to $T_{GPU\text{-}Frame1}$, i.e., the predicted latency of a current frame is equal to the latency of a previous frame. Also, the headroom is equal to the VSYNC time period

410 subtracted by (T_{qb}-T_{vsync}). Accordingly, the headroom is the amount of room left between the following VSYNC timestamp, e.g., third VSYNC time 423, and the frame ready timestamp, e.g., (T_{qb}-T_{vsync}). In some aspects, if the VSYNC time period added to the headroom is less than the predicted latency of a current frame, then the GPU frequency may be increased or boosted in order to avoid a potential future jank. As such, if VSYNC time period 410 + headroom < $T_{gpu\text{-}predict}$, this means a jank may occur in next frame, so aspects of the present disclosure can boost the GPU frequency to avoid potential janks. Also, if (T_{qb}-T_{vsync}) + $T_{gpu\text{-}predict}$ is within one VSYNC time period, then the present disclosure may not adjust the GPU frequency. However, if (T_{qb}-T_{vsync}) + $T_{gpu\text{-}predict}$ is beyond one VSYNC time period, then the GPU frequency may be boosted or increased. By detecting if [(T_{qb}-T_{vsync}) + $T_{gpu\text{-}predict}$] is beyond one VSYNC time period, the present disclosure may boost the GPU frequency to avoid a jank.

[0054] In some aspects, reducing the GPU latency for a current frame, e.g., frame 402, may equate with boosting the GPU frequency for that frame, such that the GPU execution or latency duration is reduced. For example, the execution duration for frame 402 may be less than the predicted latency, e.g., the GPU execution duration for frame 401, because the GPU frequency was boosted. After the frequency is boosted for a particular frame, e.g., frame 402, and the frame execution or latency duration is reduced, that particular frame may complete rendering prior to the next VSYNC timestamp. By completing the frame rendering prior to the next VSYNC timestamp, the present disclosure may avoid a potential jank and/or reduce the chances of any future janks.

[0055] As mentioned above, if a GPU will not finish rendering a current frame, e.g., frame 402, by the following VSYNC signal, e.g., fourth VSYNC time 424, then the GPU frequency may be boosted for the current frame. In some aspects, this GPU frequency boost may be based on the previous frame execution or latency duration, or the predicted execution or latency duration for the current frame. By doing so, the current frame may finish rendering prior to the following VSYNC timestamp, e.g., fourth VSYNC time 424. As such, aspects of the present disclosure can determine T_{qb}-T_{vsync}, i.e., the frame ready latency compared with the previous VSYNC signal, and compare it to the latency or execution time period for a previous frame.

[0056] FIG. 5 illustrates an example timing diagram 500 in accordance with one or more techniques of this disclosure. As shown in FIG. 5, timing diagram 500 includes first

frame 501, second frame 502, third frame 503, fourth frame 504, fifth frame 505, sixth frame 506, VSYNC time period (VSYNC_period) 510, first VSYNC time 521, second VSYNC time 522, third VSYNC time 523, fourth VSYNC time 524, fifth VSYNC time 525, sixth VSYNC time 526, seventh VSYNC time 527, and eighth VSYNC time 528. FIG. 5 also displays a renderer, a user interface (UI), a GPU, a frame composer or SF mechanism, a display or display engine, and a VSYNC timing mechanism. The display or display engine can also be referred to as other terms, such as a display buffer. Additionally, FIG. 5 displays a frame offset time that is equal to a difference between a frame rendering completion time and a previous VSYNC time. For example, the frame offsets for each frame can be: frame offset time T1 for frame 503, frame offset time T2 for frame 504, frame offset time T3 for frame 505, and frame offset time T4 for frame 506.

[0057] As shown in FIG. 5, aspects of the present disclosure can utilize jank reduction technology based on a predicted GPU execution or latency duration. As mentioned above, the predicted GPU execution or latency duration for a current frame can be based on a GPU execution or latency duration for a previous frame. For instance, a threshold to boost a GPU frequency can dynamically change based on a GPU execution or latency duration for a previous frame. In some instances, aspects of the present disclosure can determine the GPU frame execution or latency duration for a previous frame, e.g., $T_{gpu\text{-}predict}$ for frame 502, in order to predict the GPU frame execution or latency duration for a current frame, e.g., frame 503.

[0058] As indicate herein, aspects of the present disclosure can determine a previous frame execution or latency duration, e.g., $T_{gpu\text{-}predict}$ for frame 502, add this to a frame offset time, e.g., T1 for frame 503, and compare this value to a VSYNC time period, e.g., VSYNC_period 510. As indicated above, the frame offset time, e.g., T1, can be the difference between a previous VSYNC timestamp, e.g., fourth VSYNC time 524, and a frame rendering completion time, e.g., when the renderer completes rendering frame 503. In some aspects, if the sum of the frame offset time and the previous frame execution or latency duration is less than or equal to a VSYNC time period, then the frame composer or SF mechanism can be triggered directly and a GPU frequency boost may not be needed. As mentioned above, when the buffer queue is ready, the SF mechanism can be triggered directly. As such, if $T1 + T_{gpu\text{-}predict} \leq$ VSYNC_period 510, then SF mechanism is directly triggered. However, if the sum of the frame offset time and the previous frame execution or latency duration is greater than a VSYNC

time period, then the GPU frequency can be increased to speed up the frame execution duration. Indeed, if T1 + $T_{gpu\text{-}predict}$ > VSYNC_period 510, then the GPU frequency boost may be triggered.

[0059] As shown in FIG. 5, frames 503, 504, and 505 may not need a GPU frequency boost because the frame offset time and the previous frame execution is less than or equal to a VSYNC time period. For frame 503, T1 + frame 502's execution duration is less than VSYNC_period 510. For frame 504, T2 + frame 503's execution duration is less than VSYNC_period 510. For frame 505, T3 + frame 504's execution duration is less than VSYNC_period 510. As mentioned above, the frame offset time is the delay between the frame rendering completion time and the previous VSYNC time. If the frame offset time plus the previous frame latency duration extends beyond an entire VSYNC period, then the GPU frequency will be boosted.

[0060] As shown in FIG. 5, frame 506 may need a GPU frequency boost as the frame offset time plus the previous frame execution duration extends beyond an entire VSYNC period. For example, T4 + frame 505's execution duration is greater than VSYNC_period 510. Accordingly, the GPU frequency boost is triggered as frame 506 would not complete rendering within VSYNC_period 510. By triggering the GPU frequency boost, the GPU can execute frame 506 at an increased speed, such that any potential janks can be avoided.

[0061] As mentioned above, aspects of the present disclosure can utilize a dynamic threshold calculation in order to reduce any potential janks. In some aspects, a dynamic threshold calculation method can include determining the previous GPU execution duration and marking it as $T_{gpu\text{-}predict}$. The dynamic threshold calculation method can also include determining the buffer queue call timestamp of a current frames and marking it as T_{qb}. The dynamic threshold calculation method can also include determining the latest VSYNC timestamp and marking it as T_{vsync}. The dynamic threshold calculation method can also include triggering a scheduled GPU boost when a potential jank is detected in a SF mechanism. In some aspects, this can result in some tasks being moved to a certain processing unit in the CPU, e.g., a big cluster unit, in order to speed up the rendering process. Also, the dynamic threshold calculation method can include setting a threshold T, wherein T = VSYNC_period – $T_{gpu\text{-}predict}$ when a potential jank is detected. The dynamic threshold calculation method can also include bypassing a VSYNC time if $T_{qb} - T_{vsync} \leq T$, and triggering an SF mechanism to perform the frame composition. In FIG. 5, frames 503, 504, and

505 match this condition. Also, the dynamic threshold calculation method can include bypassing a VSYNC time if $T_{qb} - T_{vsync} > T$ and $T_{qb} - T_{vsync} < T + T_{boost}$, and triggering an SF mechanism to perform the frame composition with a GPU frequency boost for the current frame. Here, T_{boost} stands for the boost threshold, which can have a fixed or maximum boost value or be dynamically tuned. In FIG. 5, frame 506 matches this condition.

[0062] As shown in FIG. 5, aspects of the present disclosure can determine whether a current frame, e.g., frame 503, completes rendering after a first VSYNC time, e.g., fourth VSYNC time 524. Aspects of the present disclosure can also calculate a previous frame GPU execution duration, e.g., $T_{gpu\text{-}predict}$ for frame 502. Also, aspects of the present disclosure can determine a current frame offset duration, e.g., T1, when a current frame rendering completion time, e.g., when renderer complete rendering frame 503, is after the first VSYNC time, e.g., fourth VSYNC time 524. In some aspects, the current frame offset duration, e.g., T1, can be equal to a difference between the first VSYNC time, fourth VSYNC time 524, and the current frame rendering completion time, e.g., frame 503 rendering completion time.

[0063] Additionally, as shown in FIG. 5, aspects of the present disclosure can determine whether a sum of a previous frame GPU execution duration, e.g., $T_{gpu\text{-}predict}$ for frame 502, and the current frame offset duration, e.g., T1, is less than or equal to a first VSYNC period, e.g., VSYNC_period 510. In some aspects, the first VSYNC period, e.g., VSYNC_period 510, can begin at the first VSYNC time, e.g., fourth VSYNC time 524, and end at a second VSYNC time, e.g., fifth VSYNC time 525. Aspects of the present disclosure can also execute a current frame, e.g., frame 503, based on the determination of whether the sum of the previous frame GPU execution duration, e.g., $T_{gpu\text{-}predict}$ for frame 502, and the current frame offset duration, e.g., T1, is less than or equal to the first VSYNC period, e.g., VSYNC_period 510. In some aspects, the current frame, e.g., frame 503, can be executed in a current frame GPU execution duration that begins at a current frame execution start time and ends at a current frame execution completion time.

[0064] Moreover, as shown in FIG. 5, aspects of the present disclosure can also reduce the current frame GPU execution duration, e.g., the duration of GPU executing frame 503, when the sum of the previous frame GPU execution duration, e.g., $T_{gpu\text{-}predict}$ for frame 502, and the current frame offset duration, e.g., T1, is greater than the first VSYNC period, e.g., VSYNC_period 510. Aspects of the present disclosure can also

increase a frame execution frequency when the sum of the previous frame GPU execution duration, e.g., $T_{gpu\text{-}predict}$ for frame 502, and the current frame offset duration, e.g., T1, is greater than the first VSYNC period, e.g., VSYNC_period 510. In some aspects, the frame execution frequency can be increased over the current frame GPU execution duration, e.g., the duration of GPU executing frame 503. Additionally, the frame execution frequency can be increased by a GPU. In some aspects, the previous frame GPU execution duration, e.g., $T_{gpu\text{-}predict}$ for frame 502, can be equal to a predicted current frame GPU execution duration. Also, the current frame offset duration, e.g., T1, can be equal to a difference between the first VSYNC time, e.g., fourth VSYNC time 524, and the current frame execution start time, e.g., when GPU starts executing frame 503.

[0065] In addition, as shown in FIG. 5, aspects of the present disclosure can consume the current frame, e.g., frame 503, based on the current frame offset duration, e.g., T1, being greater than, less than, or equal to a threshold, e.g., T = VSYNC_period 510 – $T_{gpu\text{-}predict}$. For example, aspects of the present disclosure can consume the current frame, e.g., frame 503, at the current frame rendering completion time when the current frame offset duration, e.g., T1, is less than or equal to a threshold, e.g., T = VSYNC_period 510 – $T_{gpu\text{-}predict}$. In some aspects, a SF mechanism can consume the current frame, e.g., frame 503, at the current frame rendering completion time. Also, aspects of the present disclosure can consume the current frame, e.g., frame 503, at the first VSYNC time, e.g., fourth VSYNC time 524, when the current frame rendering completion time is before or equal to the first VSYNC time, e.g., fourth VSYNC time 524. Further, aspects of the present disclosure can consume the current frame, e.g., frame 503, at the second VSYNC time, e.g., fifth VSYNC time 525, when the current frame offset duration, e.g., T1, is greater than a threshold, e.g., T = VSYNC_period 510 – $T_{gpu\text{-}predict}$. Additionally, aspects of the present disclosure can send the current frame, e.g., frame 503, to a buffer queue at the current frame rendering completion time. Also, aspects of the present disclosure can increase a buffer queue counter of the buffer queue when the current frame, e.g., frame 503, is sent to the buffer queue.

[0066] FIG. 6 illustrates an example timing diagram 600 in accordance with one or more techniques of this disclosure. As shown in FIG. 6, timing diagram 600 includes first frame 601, second frame 602, VSYNC time period (VSYNC_period) 610, first VSYNC time 621, second VSYNC time 622, and third VSYNC time 623. FIG. 6 also

displays a GPU, a frame composer or SF mechanism, a SF binder, and a buffer queue. FIG. 6 also displays one aspect of jank reduction technology based on a GPU execution or latency duration prediction.

[0067] As shown in FIG. 6, the predicted execution or latency duration for frame 602 is based on the execution or latency duration for frame 601, e.g., $T_{GPU\text{-}Frame1}$. This $T_{GPU\text{-}Frame1}$ value plus the time between the previous VSYNC time 622 and the start of the rendering for frame 602 (T_{qb}-T_{vsync}) is compared to the VSYNC time period 610. As shown in FIG. 6, $T_{GPU\text{-}Frame1}$ + (T_{qb}-T_{vsync}) is less than VSYNC time period 610, so there is no need to boost the GPU frequency or the CPU.

[0068] In some aspects, when aspects of the present disclosure detect there is no available frame in the buffer queue, i.e., frame 602 is not ready, then the scheduled boost can be triggered. Accordingly, the present disclosure can prepare for a possible GPU boost. However, if (T_{qb}-T_{vsync}) + $T_{gpu\text{-}predict}$ is less than the VSYNC time period 610, then the boost will not be utilized. As such, aspects of the present disclosure can prepare for a possible GPU boost, but not actually utilize the boost.

[0069] As shown in FIG. 6, when frame 602 completes rendering, the present disclosure can obtain a delay parameter to get a scale parameter, e.g., in order to ascertain how may CPU upscaling parameters may need to be set. When the scale parameter is obtained, the minimum CPU frequency can be set. In some aspects, on the CPU side, when a possible jank is detected, some tasks can be moved to certain processing units, e.g., a big cluster unit, in order to speed up the rendering process. When the frame finishes rendering, aspects of the present disclosure can obtain one scale parameter to renew a minimum CPU frequency for each CPU cluster. In some instances, this can provide a benefit to the next frame latency duration.

[0070] In some aspects, the scheduled boost can be triggered by the SF mechanism based on a possible future jank. Also, for the scheduled boost, a scale parameter may not be used, as scaling may not be needed on the CPU side. When a frame finishes rendering, aspects of the present disclosure can determine the scale parameter in order to determine the scaling for the next frame. In some aspects, e.g., when there is no GPU boost, the scaling may be performed in the CPU side. For example, as shown in FIG. 6, the scaling parameter can be equal to VSYNC_period / (VSYNC_period – (T_{qb}-T_{vsync})). In some instances, this scaling calculation may be performed on the CPU side.

[0071] As mentioned herein, when a jank is detected, the scheduled boost can be triggered by the SF mechanism, and some tasks can be moved to a certain processing unit, e.g., a big cluster unit, in order to speed up the frame rendering. At the same time, the CPU frequency upscaling can occur, e.g., based on the value (T_{qb}-T_{vsync}). In some aspects, a certain processing unit's frequency can be updated, e.g., a big cluster's CPU minimum frequency may be updated, based on a certain calculation, e.g., calcCpuxMinFreq = cpuxCurrFreq * scaling value. As mentioned above, $T_{gpu\text{-}predict}$ can be equal to $T_{GPU\text{-}Frame1}$. Also, the dynamic threshold for frame 602 can be T, wherein T = VSYNC_period - $T_{gpu\text{-}predict}$, e.g., when $T_{gpu\text{-}predict}$ < VSYNC_period. Further, if T_{qb}-T_{vsync} < T, then the SF mechanism can be triggered to perform the frame composition, e.g., and bypass the VSYNC signal, such that there may be no need to boost the GPU and/or CPU.

[0072] FIG. 7 illustrates an example timing diagram 700 in accordance with one or more techniques of this disclosure. As shown in FIG. 7, timing diagram 700 includes first frame 701, second frame 702, VSYNC time period (VSYNC_period) 710, first VSYNC time 721, second VSYNC time 722, and third VSYNC time 723. FIG. 7 also displays a GPU, a frame composer or SF mechanism, a SF binder, and a buffer queue. FIG. 7 also displays one aspect of jank reduction technology based on a GPU execution or latency duration prediction.

[0073] As shown in FIG. 7, timing diagram 700 is similar to timing diagram 600 above, except a GPU boost may occur for the execution of frame 702. Accordingly, the execution of frame 702 is shown in gray rather than white. As shown in FIG. 7, when a potential jank is detected at the SF mechanism, the SF mechanism can determine that there is no available frame in the buffer queue at the VSYNC time 722, e.g., frame 702 is not ready for rendering. So when a potential jank is detected, the boost may be triggered by the SF mechanism, and some CPU tasks may be moved to a certain processing unit, e.g., a big cluster unit, to speed up the rendering.

[0074] In some aspects, in order to calculate if frame 702 will finish rendering before the next VSYNC time 723, the previous frame GPU execution duration for frame 701, e.g., $T_{GPU\text{-}Frame1}$, can be used to predict the frame GPU execution duration for frame 702. Also, the scale parameter equal to VSYNC_period / (VSYNC_period – (T_{qb}-T_{vsync})) can be calculated on the CPU side, e.g., for CPU scaling. On the GPU side, when the present disclosure determines a GPU boost may be needed, GPU frequency can be increased to the maximum frequency for the current frame. In some aspects,

the duration can be about one frame latency duration, e.g., for 60 FPS mode the duration can be 16 ms. In some aspects, in order to boost the GPU, the present disclosure can increase the GPU frequency to speed up the frame execution time. So in some aspects, if the present disclosure determines that the next frame cannot finish rendering prior to the next VSYNC time, then the GPU frequency can be boosted to the maximum frequency. Further, the boost duration can be one frame period, which for 60 FPS mode can be about 16 ms, and for 90 FPS mode can be about 11 ms.

[0075] As indicated herein, aspects of the present disclosure can calculate the frame offset period from a first VSYNC time until a frame rendering completion time for a current frame. Then, the present disclosure can add the frame offset period to the predicted execution time for the current frame, e.g., based on the previous frame's GPU execution duration. If the sum of the frame offset period and the predicted latency time is less than or equal to one VSYNC period, then the frame can be executed in a normal fashion. However, if the sum of the frame offset period and the predicted latency time is greater than one VSYNC period, then the GPU frequency can be boosted or increased in order to speed up the frame execution time, as well as avoid any potential janks.

[0076] In some aspects, when a jank is detected, a GPU boost can be triggered by an SF mechanism, and at least some tasks may be moved to a certain processing unit, e.g., a big cluster unit, in order to speed up the execution or rendering. As mentioned above, the predicted execution or latency for a current frame can be equal to a previous frame's execution or latency time, e.g., $T_{gpu\text{-}predict} = T_{GPU\text{-}Frame1}$. Also, a dynamic threshold for a current frame can be T, wherein T = VSYNC_period - $T_{gpu\text{-}predict}$, e.g., when $T_{gpu\text{-}predict}$ is less than VSYNC_period. Also, if T_{qb}-T_{vsync} is greater than or equal to T, then the SF mechanism can be triggered to perform the frame composition and bypass the VSYNC signal and boost the GPU to a maximum frequency. By doing so, aspects of the present disclosure can utilize dynamic change to avoid any potential janks for a current frame.

[0077] In some aspects, by utilizing the predicted frame latency described above, aspects of the present disclosure can reduce the amount of janks, e.g., by up to 67%. For example, in some instances, aspects of the present disclosure can reduce the amount of janks experienced in a mobile gaming application over a time period from 131 janks to 43 janks. In other aspects, the amount of janks can be reduced by 50%. Also, with the dynamic threshold optimization described herein, the FPS data can also

improve. For example, the FPS can improve by 2.5%. Further, the jank latency can be reduced by utilizing the methods herein, e.g., from 117 ms to 83 ms. In some aspects, the present disclosure can reduce the amount of janks, as well as eliminate any janks with a large latency, e.g., janks with a latency of about 117 ms. By utilizing the jank reduction optimization methods herein, aspects of the present disclosure can help to improve the overall user experience.

[0078] FIG. 8 illustrates an example flowchart 800 of an example method in accordance with one or more techniques of this disclosure. The method may be performed by a frame composer, display processor, GPU, or apparatus for graphics processing. At 802, the apparatus can determine whether a current frame completes rendering after a first VSYNC time, as described in connection with the examples in FIGs. 3, 4, 5, 6, and 7. At 804, the apparatus can calculate a previous frame GPU execution duration, as described in connection with the examples in FIGs. 3, 4, 5, 6, and 7.

[0079] At 806, the apparatus can determine a current frame offset duration when a current frame rendering completion time is after the first VSYNC time, as described in connection with the examples in FIGs. 3, 4, 5, 6, and 7. In some aspects, the current frame offset duration can be equal to a difference between the first VSYNC time and the current frame rendering completion time, as described in connection with the examples in FIGs. 3, 4, 5, 6, and 7. At 808, the apparatus can also determine whether a sum of a previous frame GPU execution duration and the current frame offset duration is less than or equal to a first VSYNC period, as described in connection with the examples in FIGs. 3, 4, 5, 6, and 7. In some aspects, the first VSYNC period can begin at the first VSYNC time and end at a second VSYNC time, as described in connection with the examples in FIGs. 3, 4, 5, 6, and 7. At 810, the apparatus can execute a current frame based on the determination of whether the sum of the previous frame GPU execution duration and the current frame offset duration is less than or equal to the first VSYNC period, as described in connection with the examples in FIGs. 3, 4, 5, 6, and 7. In some aspects, the current frame can be executed in a current frame GPU execution duration that begins at a current frame execution start time and ends at a current frame execution completion time, as described in connection with the examples in FIGs. 3, 4, 5, 6, and 7.

[0080] At 812, the apparatus can also reduce the current frame GPU execution duration when the sum of the previous frame GPU execution duration and the current frame offset duration is greater than the first VSYNC period, as described in connection with the

examples in FIGs. 3, 4, 5, 6, and 7. At 814, the apparatus can increase a frame execution frequency when the sum of the previous frame GPU execution duration and the current frame offset duration is greater than the first VSYNC period, as described in connection with the examples in FIGs. 3, 4, 5, 6, and 7. In some aspects, the frame execution frequency can be increased over the current frame GPU execution duration, as described in connection with the examples in FIGs. 3, 4, 5, 6, and 7. Additionally, the frame execution frequency can be increased by a GPU, as described in connection with the examples in FIGs. 3, 4, 5, 6, and 7. In some aspects, the previous frame GPU execution duration can be equal to a predicted current frame GPU execution duration, as described in connection with the examples in FIGs. 3, 4, 5, 6, and 7. Also, the current frame offset duration can be equal to a difference between the first VSYNC time and the current frame execution start time, as described in connection with the examples in FIGs. 3, 4, 5, 6, and 7.

[0081] At 816, the apparatus can consume the current frame based on the current frame offset duration being greater than, less than, or equal to a threshold, as described in connection with the examples in FIGs. 3, 4, 5, 6, and 7. For example, the apparatus can consume the current frame at the current frame rendering completion time when the current frame offset duration is less than or equal to a threshold, as described in connection with the examples in FIGs. 3, 4, 5, 6, and 7. In some aspects, a surface flinger mechanism can consume the current frame at the current frame rendering completion time, as described in connection with the examples in FIGs. 3, 4, 5, 6, and 7. Also, the apparatus can consume the current frame at the first VSYNC time when the current frame rendering completion time is before or equal to the first VSYNC time, as described in connection with the examples in FIGs. 3, 4, 5, 6, and 7. Further, the apparatus can consume the current frame at the second VSYNC time when the current frame offset duration is greater than a threshold, as described in connection with the examples in FIGs. 3, 4, 5, 6, and 7.

[0082] At 818, the apparatus can send the current frame to a buffer queue at the current frame rendering completion time, as described in connection with the examples in FIGs. 3, 4, 5, 6, and 7. At 820, the apparatus can increase a buffer queue counter of the buffer queue when the current frame is sent to the buffer queue, as described in connection with the examples in FIGs. 3, 4, 5, 6, and 7.

[0083] In one configuration, a method or apparatus for graphics processing is provided. The apparatus may be a GPU or some other processor that can perform graphics

processing. In one aspect, the apparatus may be the processing unit 120 within the device 104, or may be some other hardware within device 104 or another device. The apparatus may include means for determining a current frame offset duration when a current frame rendering completion time is after a first VSYNC time. The apparatus may also include means for determining whether a sum of a previous frame GPU execution duration and the current frame offset duration is less than or equal to a first VSYNC period. The apparatus may also include means for executing a current frame based on the determination of whether the sum of the previous frame GPU execution duration and the current frame offset duration is less than or equal to the first VSYNC period. The apparatus may also include means for reducing the current frame GPU execution duration when the sum of the previous frame GPU execution duration and the current frame offset duration is greater than the first VSYNC period. The apparatus may also include means for increasing a frame execution frequency when the sum of the previous frame GPU execution duration and the current frame offset duration is greater than the first VSYNC period. The apparatus may also include means for determining whether a current frame completes rendering after the first VSYNC time. The apparatus may also include means for consuming the current frame at the current frame rendering completion time when the current frame offset duration is less than or equal to a threshold. The apparatus may also include means for consuming the current frame at the first VSYNC time when the current frame rendering completion time is before or equal to the first VSYNC time. The apparatus may also include means for consuming the current frame at the second VSYNC time when the current frame offset duration is greater than a threshold. The apparatus may also include means for sending the current frame to a buffer queue at the current frame rendering completion time. The apparatus may also include means for increasing a buffer queue counter of the buffer queue when the current frame is sent to the buffer queue. The apparatus may also include means for calculating the previous frame GPU execution duration.

[0084] The subject matter described herein can be implemented to realize one or more benefits or advantages. For instance, the described graphics processing techniques can be used by GPUs, frame composers, display processors, or other processors to enable increased accuracy of jank detection. This can also be accomplished at a low cost compared to other graphics processing techniques. Moreover, the graphics processing techniques herein can improve or speed up data processing or execution.

Further, the graphics processing techniques herein can improve a GPU's resource or data utilization and/or resource efficiency. Additionally, aspects of the present disclosure can utilize jank reduction technology including a dynamic threshold, which can result in janks prediction with increased accuracy and reduced power consumption.

[0085] In accordance with this disclosure, the term "or" may be interrupted as "and/or" where context does not dictate otherwise. Additionally, while phrases such as "one or more" or "at least one" or the like may have been used for some features disclosed herein but not others, the features for which such language was not used may be interpreted to have such a meaning implied where context does not dictate otherwise.

[0086] In one or more examples, the functions described herein may be implemented in hardware, software, firmware, or any combination thereof. For example, although the term "processing unit" has been used throughout this disclosure, such processing units may be implemented in hardware, software, firmware, or any combination thereof. If any function, processing unit, technique described herein, or other module is implemented in software, the function, processing unit, technique described herein, or other module may be stored on or transmitted over as one or more instructions or code on a computer-readable medium. Computer-readable media may include computer data storage media or communication media including any medium that facilitates transfer of a computer program from one place to another. In this manner, computer-readable media generally may correspond to (1) tangible computer-readable storage media, which is non-transitory or (2) a communication medium such as a signal or carrier wave. Data storage media may be any available media that can be accessed by one or more computers or one or more processors to retrieve instructions, code and/or data structures for implementation of the techniques described in this disclosure. By way of example, and not limitation, such computer-readable media can comprise RAM, ROM, EEPROM, CD-ROM or other optical disk storage, magnetic disk storage or other magnetic storage devices,. Disk and disc, as used herein, includes compact disc (CD), laser disc, optical disc, digital versatile disc (DVD), floppy disk and Blu-ray disc where disks usually reproduce data magnetically, while discs reproduce data optically with lasers. Combinations of the above should also be included within the scope of computer-readable media. A computer program product may include a computer-readable medium.

[0087] The code may be executed by one or more processors, such as one or more digital signal processors (DSPs), general purpose microprocessors, application specific integrated circuits (ASICs), arithmetic logic units (ALUs), field programmable logic arrays (FPGAs), or other equivalent integrated or discrete logic circuitry. Accordingly, the term "processor," as used herein may refer to any of the foregoing structure or any other structure suitable for implementation of the techniques described herein. Also, the techniques could be fully implemented in one or more circuits or logic elements.

[0088] The techniques of this disclosure may be implemented in a wide variety of devices or apparatuses, including a wireless handset, an integrated circuit (IC) or a set of ICs, e.g., a chip set. Various components, modules or units are described in this disclosure to emphasize functional aspects of devices configured to perform the disclosed techniques, but do not necessarily need realization by different hardware units. Rather, as described above, various units may be combined in any hardware unit or provided by a collection of interoperative hardware units, including one or more processors as described above, in conjunction with suitable software and/or firmware.

[0089] Various examples have been described. These and other examples are within the scope of the following claims.

CLAIMS

WHAT IS CLAIMED IS:

1. A method of frame processing, comprising:

determining a current frame offset duration when a current frame rendering completion time is after a first vertical synchronization (VSYNC) time, wherein the current frame offset duration is equal to a difference between the first VSYNC time and the current frame rendering completion time;

determining whether a sum of a previous frame graphics processing unit (GPU) execution duration and the current frame offset duration is less than or equal to a first VSYNC period, wherein the first VSYNC period begins at the first VSYNC time and ends at a second VSYNC time; and

executing a current frame based on the determination of whether the sum of the previous frame GPU execution duration and the current frame offset duration is less than or equal to the first VSYNC period.

2. The method of claim 1, wherein the current frame is executed in a current frame GPU execution duration that begins at a current frame execution start time and ends at a current frame execution completion time.

3. The method of claim 2, further comprising:

reducing the current frame GPU execution duration when the sum of the previous frame GPU execution duration and the current frame offset duration is greater than the first VSYNC period.

4. The method of claim 3, wherein reducing the current frame GPU execution duration further comprises:

increasing a frame execution frequency when the sum of the previous frame GPU execution duration and the current frame offset duration is greater than the first VSYNC period.

5. The method of claim 4, wherein the frame execution frequency is increased over the current frame GPU execution duration.

6. The method of claim 4, wherein the frame execution frequency is increased by a GPU.

7. The method of claim 1, further comprising:

determining whether a current frame completes rendering after the first VSYNC time.

8. The method of claim 1, wherein the previous frame GPU execution duration is equal to a predicted current frame GPU execution duration.

9. The method of claim 1, further comprising:

consuming the current frame at the current frame rendering completion time when the current frame offset duration is less than or equal to a threshold.

10. The method of claim 9, wherein a surface flinger mechanism consumes the current frame at the current frame rendering completion time.

11. The method of claim 1, further comprising:

consuming the current frame at the first VSYNC time when the current frame rendering completion time is before or equal to the first VSYNC time.

12. The method of claim 1, further comprising:

consuming the current frame at the second VSYNC time when the current frame offset duration is greater than a threshold.

13. The method of claim 1, further comprising:

sending the current frame to a buffer queue at the current frame rendering completion time; and

increasing a buffer queue counter of the buffer queue when the current frame is sent to the buffer queue.

14. The method of claim 1, further comprising:

calculating the previous frame GPU execution duration.

15. An apparatus for frame processing, comprising:

a memory; and

at least one processor coupled to the memory and configured to:

determine a current frame offset duration when a current frame rendering completion time is after a first vertical synchronization (VSYNC) time, wherein the current frame offset duration is equal to a difference between the first VSYNC time and the current frame rendering completion time;

determine whether a sum of a previous frame graphics processing unit (GPU) execution duration and the current frame offset duration is less than or equal to a first VSYNC period, wherein the first VSYNC period begins at the first VSYNC time and ends at a second VSYNC time; and

execute a current frame based on the determination of whether the sum of the previous frame GPU execution duration and the current frame offset duration is less than or equal to the first VSYNC period.

16. The apparatus of claim 15, wherein the current frame is executed in a current frame GPU execution duration that begins at a current frame execution start time and ends at a current frame execution completion time.

17. The apparatus of claim 16, wherein the at least one processor is further configured to:

reduce the current frame GPU execution duration when the sum of the previous frame GPU execution duration and the current frame offset duration is greater than the first VSYNC period.

18. The apparatus of claim 17, wherein to reduce the current frame GPU execution duration further comprises the at least one processor is configured to:

increase a frame execution frequency when the sum of the previous frame GPU execution duration and the current frame offset duration is greater than the first VSYNC period.

19. The apparatus of claim 18, wherein the frame execution frequency is increased over the current frame GPU execution duration.

20. The apparatus of claim 18, wherein the frame execution frequency is increased by a GPU.

21. The apparatus of claim 15, wherein the at least one processor is further configured to:

determine whether a current frame completes rendering after the first VSYNC time.

22. The apparatus of claim 15, wherein the previous frame GPU execution duration is equal to a predicted current frame GPU execution duration.

23. The apparatus of claim 15, wherein the at least one processor is further configured to:

consume the current frame at the current frame rendering completion time when the current frame offset duration is less than or equal to a threshold.

24. The apparatus of claim 23, wherein a surface flinger mechanism consumes the current frame at the current frame rendering completion time.

25. The apparatus of claim 15, wherein the at least one processor is further configured to:

consume the current frame at the first VSYNC time when the current frame rendering completion time is before or equal to the first VSYNC time.

26. The apparatus of claim 15, wherein the at least one processor is further configured to:

consume the current frame at the second VSYNC time when the current frame offset duration is greater than a threshold.

27. The apparatus of claim 15, wherein the at least one processor is further configured to:

send the current frame to a buffer queue at the current frame rendering completion time; and

increase a buffer queue counter of the buffer queue when the current frame is sent to the buffer queue.

28. The apparatus of claim 15, wherein the at least one processor is further configured to:

calculate the previous frame GPU execution duration.

29. An apparatus for frame processing, comprising:

means for determining a current frame offset duration when a current frame rendering completion time is after a first vertical synchronization (VSYNC) time, wherein the current frame offset duration is equal to a difference between the first VSYNC time and the current frame rendering completion time;

means for determining whether a sum of a previous frame graphics processing unit (GPU) execution duration and the current frame offset duration is less than or equal to a first VSYNC period, wherein the first VSYNC period begins at the first VSYNC time and ends at a second VSYNC time; and

means for executing a current frame based on the determination of whether the sum of the previous frame GPU execution duration and the current frame offset duration is less than or equal to the first VSYNC period.

30. A computer-readable medium storing computer executable code for frame processing, comprising code to:

determine a current frame offset duration when a current frame rendering completion time is after a first vertical synchronization (VSYNC) time, wherein the current frame offset duration is equal to a difference between the first VSYNC time and the current frame rendering completion time;

determine whether a sum of a previous frame graphics processing unit (GPU) execution duration and the current frame offset duration is less than or equal to a first VSYNC period, wherein the first VSYNC period begins at the first VSYNC time and ends at a second VSYNC time; and

execute a current frame based on the determination of whether the sum of the previous frame GPU execution duration and the current frame offset duration is less than or equal to the first VSYNC period.

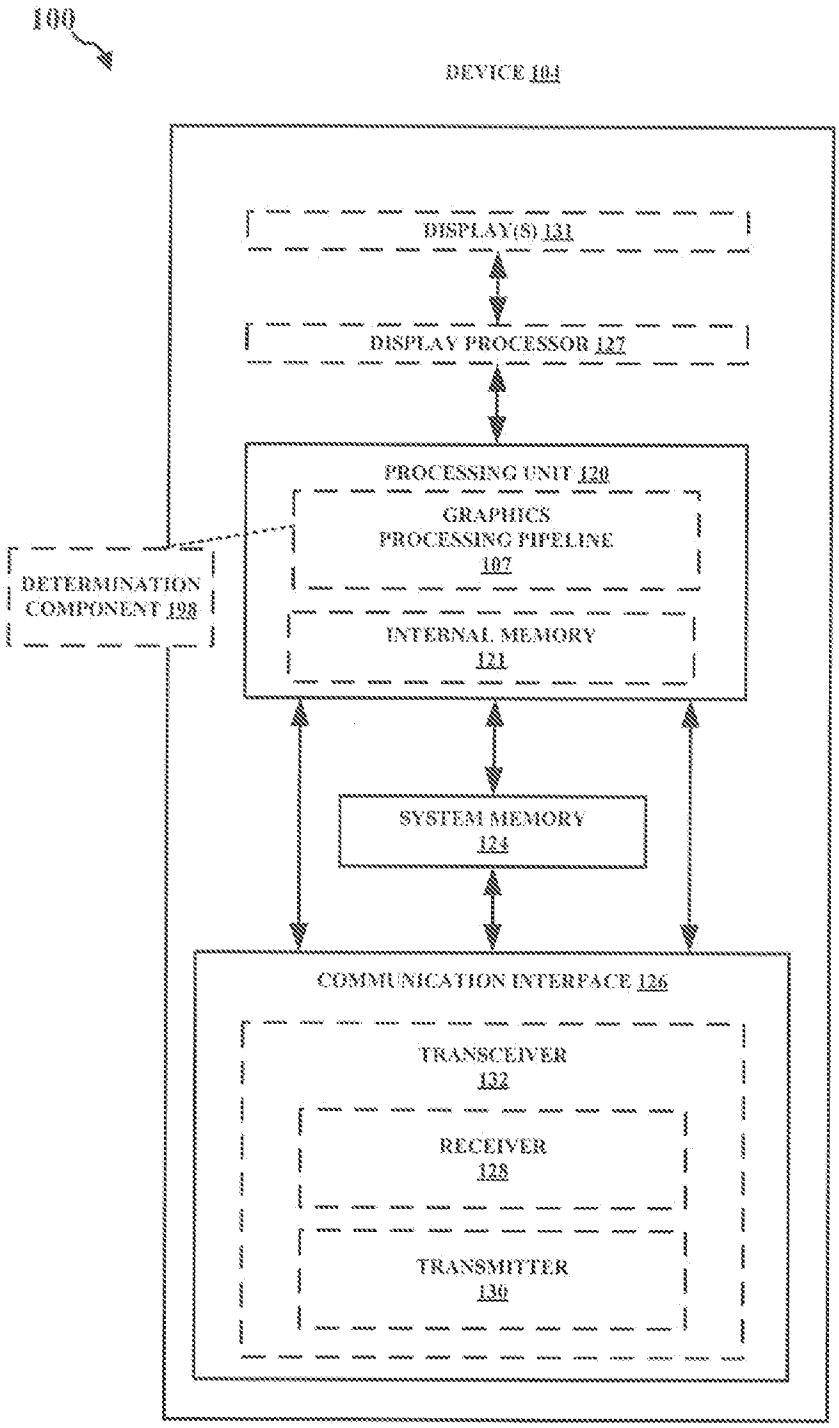
100
DEVICE 104
DISPLAY(S) 131
DISPLAY PROCESSOR 127
PROCESSING UNIT 120
GRAPHICS PROCESSING PIPELINE 107
INTERNAL MEMORY 121
DETERMINATION COMPONENT 198
SYSTEM MEMORY 124
COMMUNICATION INTERFACE 126
TRANSCEIVER 132
RECEIVER 128
TRANSMITTER 130

FIG. 1

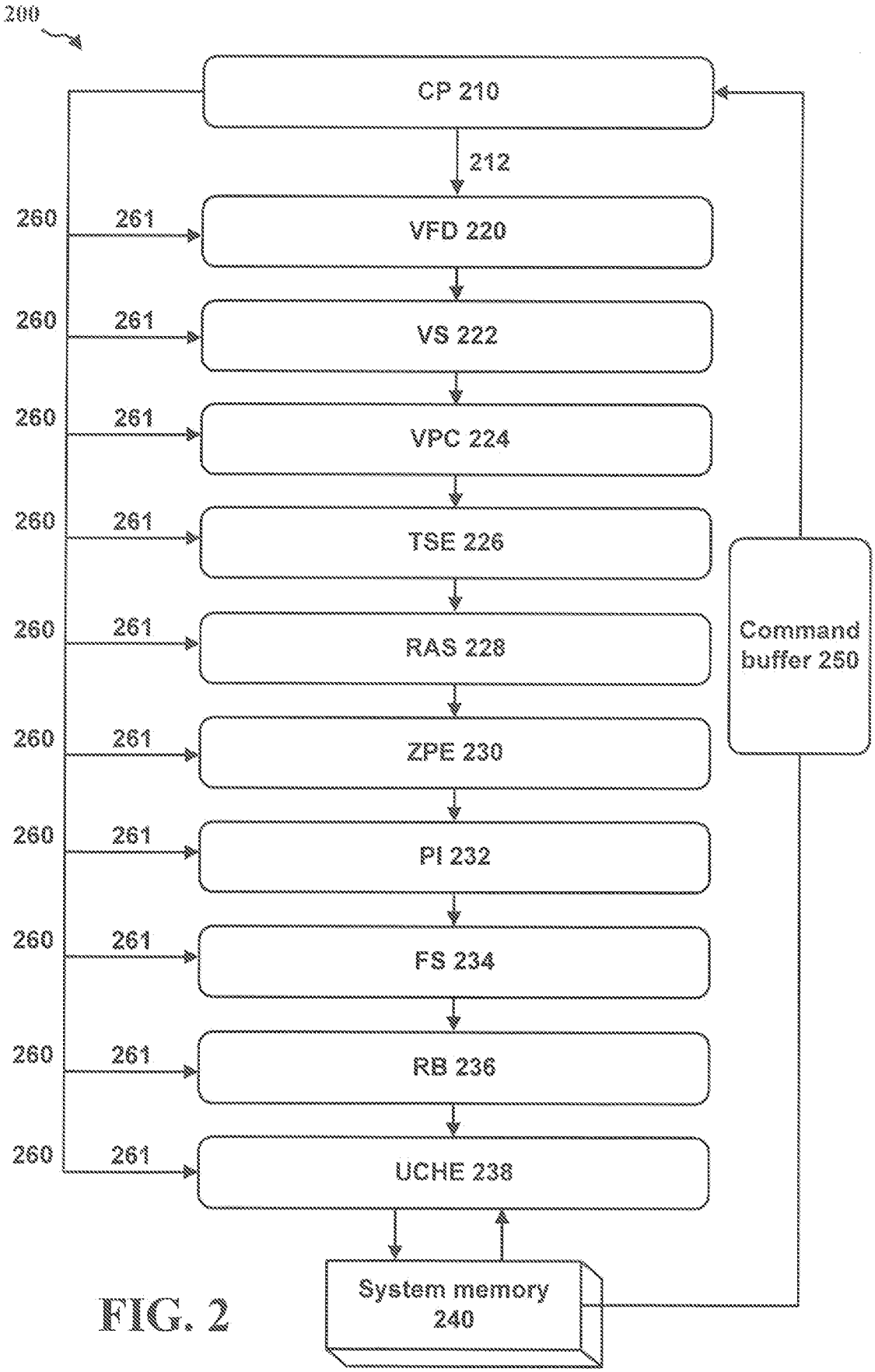
200
CP 210
212
260
261
VFD 220
VS 222
VPC 224
TSE 226
RAS 228
ZPE 230
PI 232
FS 234
RB 236
UCHE 238
Command buffer 250
System memory 240

FIG. 2

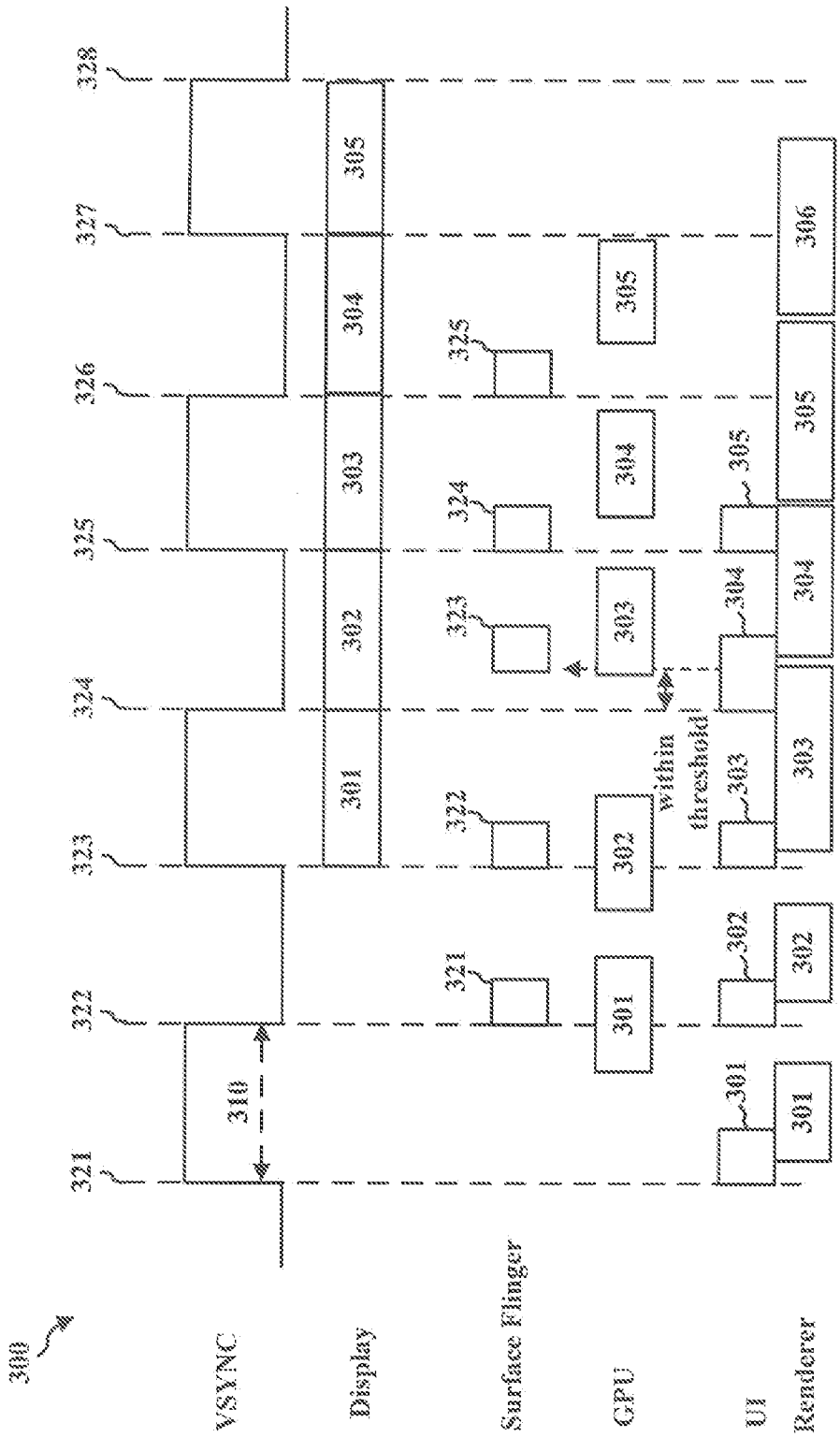
300
VSYNC
Display
Surface Flinger
GPU
UI
Renderer
321
322
323
324
325
326
327
328
310
301
302
303
304
305
306
321
322
323
324
325
301
302
303
304
305
301
302
303
304
305
306
within threshold

FIG. 3

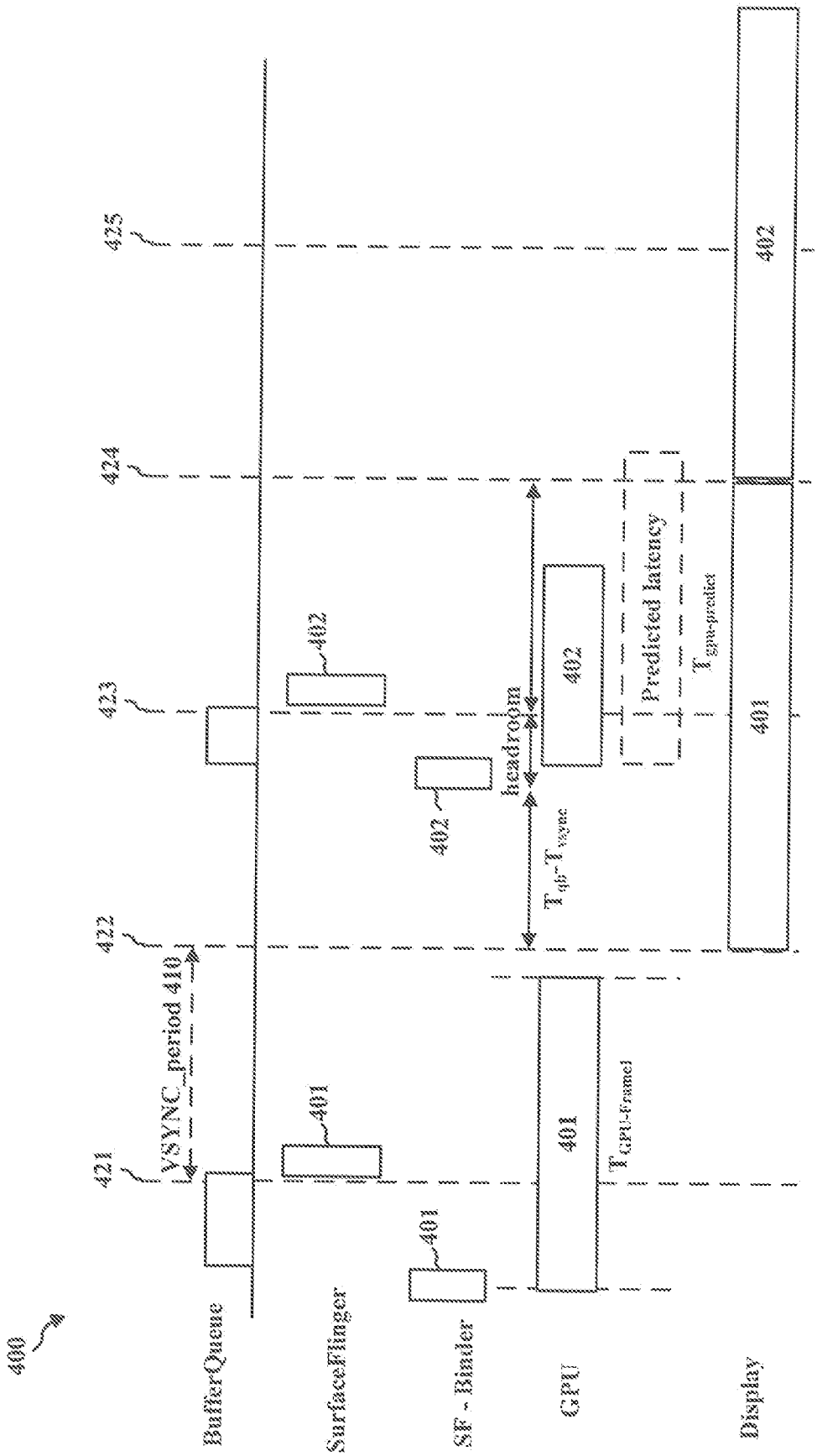
400
BufferQueue
SurfaceFlinger
SF - Binder
GPU
Display
421
422
423
424
425
VSYNC_period 410
401
402
headroom
$T_{gpu-predict}$
Predicted latency
$T_{GPU-frame1}$
T_{vsync}-T_{gpu}

FIG. 4

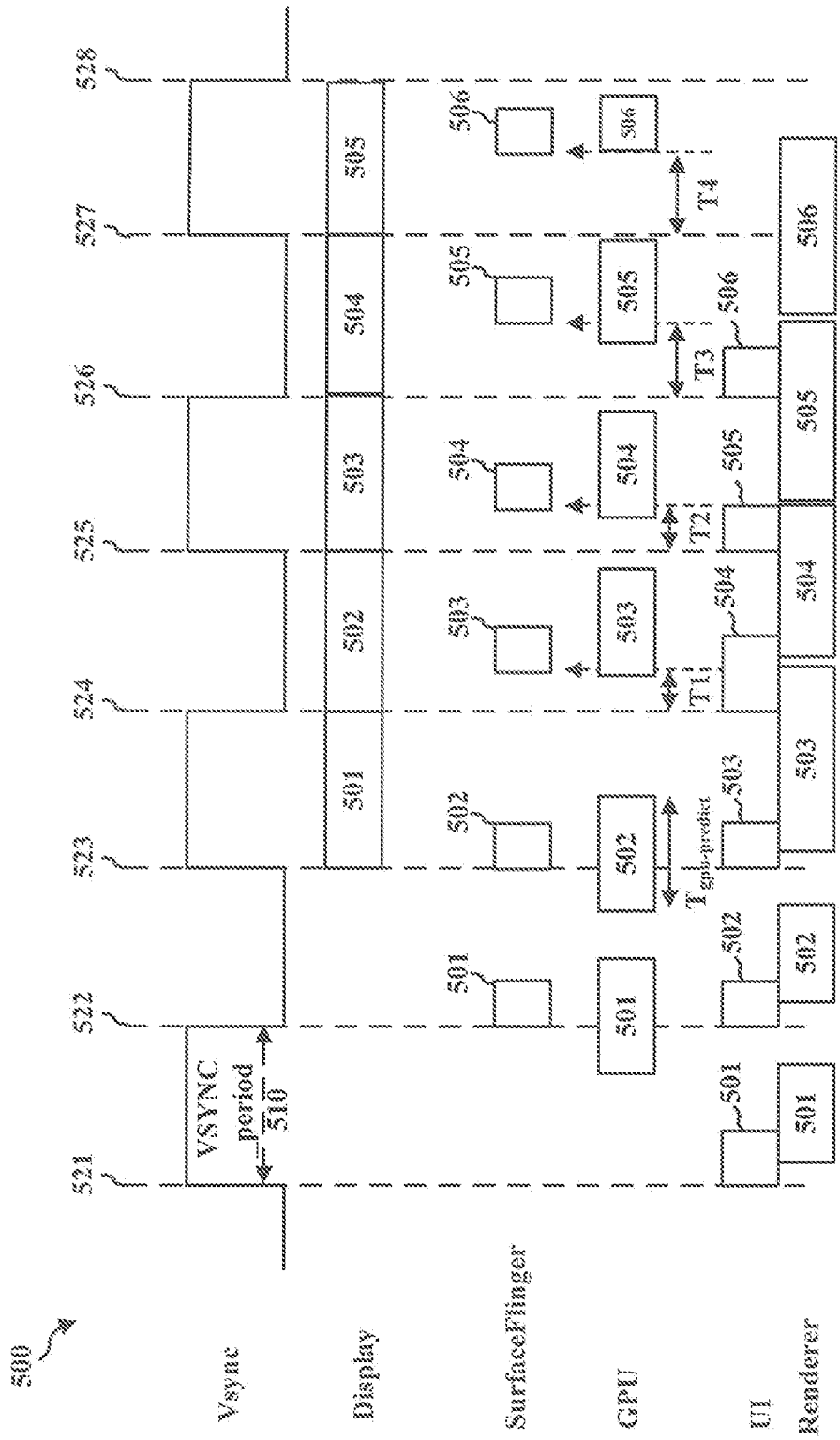
500
Vsync
Display
SurfaceFlinger
GPU
UI
Renderer
521
522
523
524
525
526
527
528
VSYNC
period
510
501
502
503
504
505
506
T1
T2
T3
T4
$T_{gpu\text{-}predict}$

FIG. 5

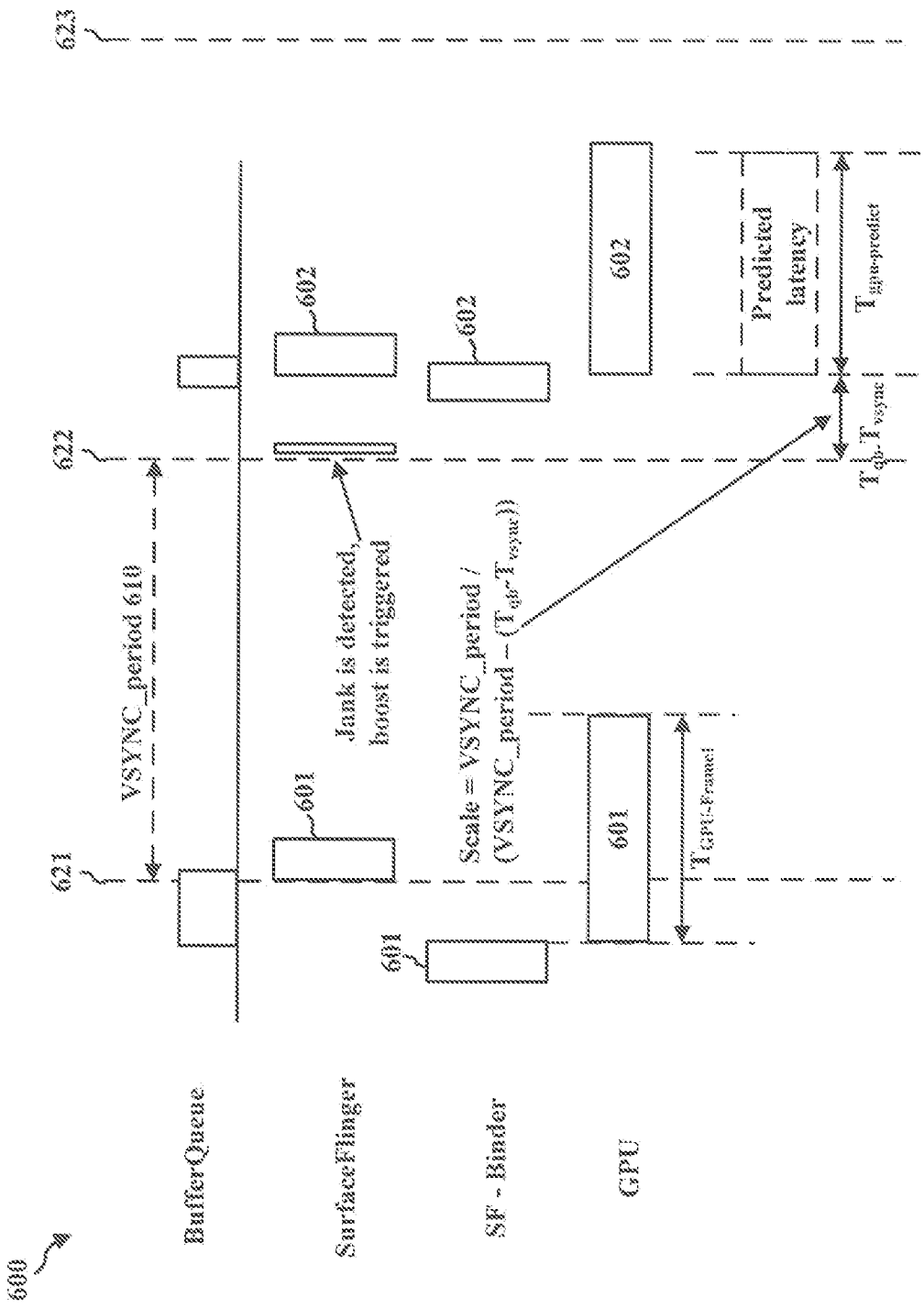
600
BufferQueue
SurfaceFlinger
SF - Binder
GPU
621
622
623
VSYNC_period 610
601
602
Jank is detected, boost is triggered
Scale = VSYNC_period / (VSYNC_period – (T_{qb}-T_{vsync}))
Predicted latency
T_{qb}
T_{vsync}
$T_{GPU-Frame}$
$T_{gpu-predict}$

FIG. 6

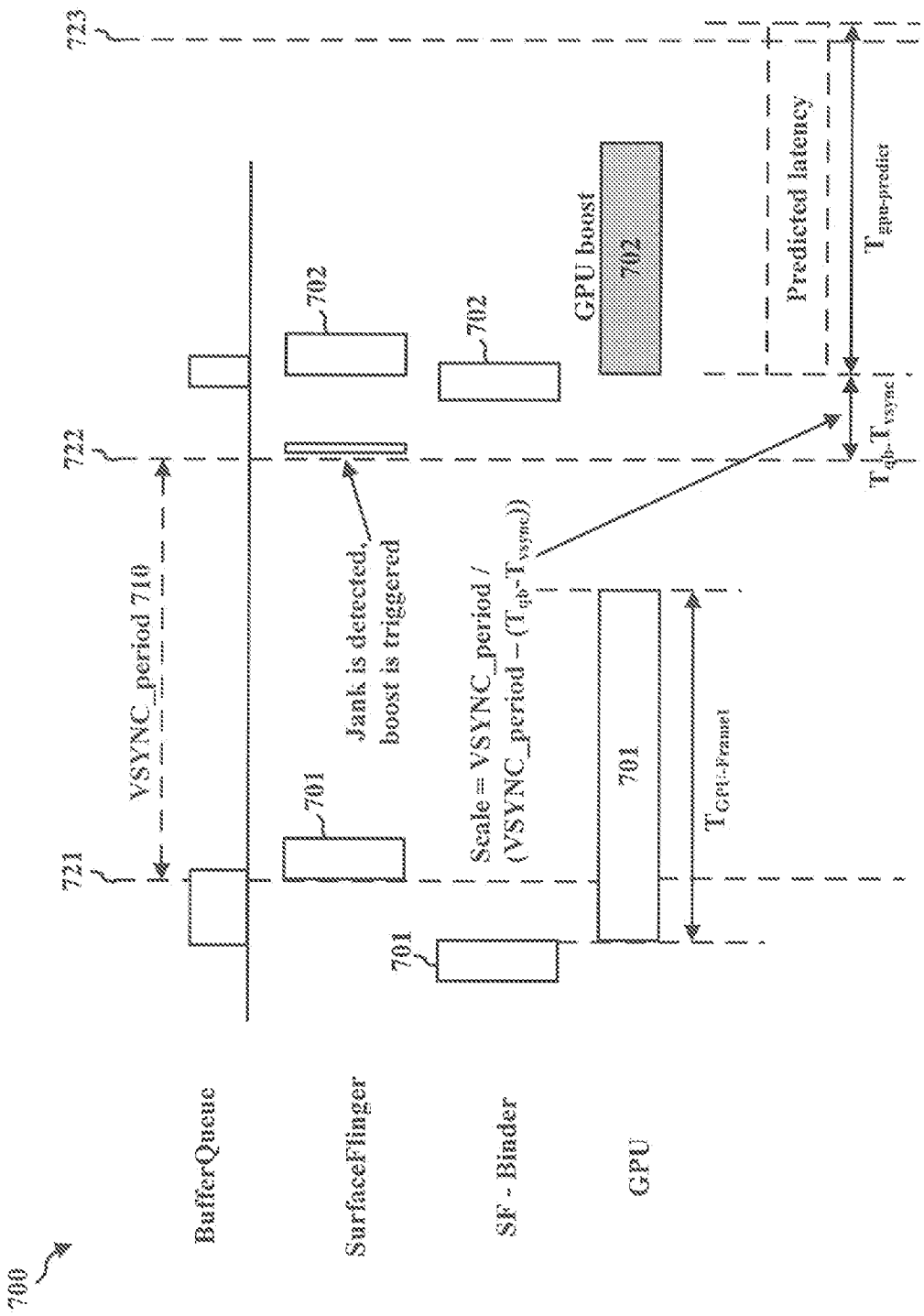
700
BufferQueue
SurfaceFlinger
SF - Binder
GPU
721
722
723
VSYNC_period 710
701
Jank is detected, boost is triggered
Scale = VSYNC_period / (VSYNC_period – (T_{qb}-T_{vsync}))
$T_{GPU-Frame}$
T_{qb}-T_{vsync}
$T_{app-predict}$
Predicted latency
GPU boost
702

FIG. 7

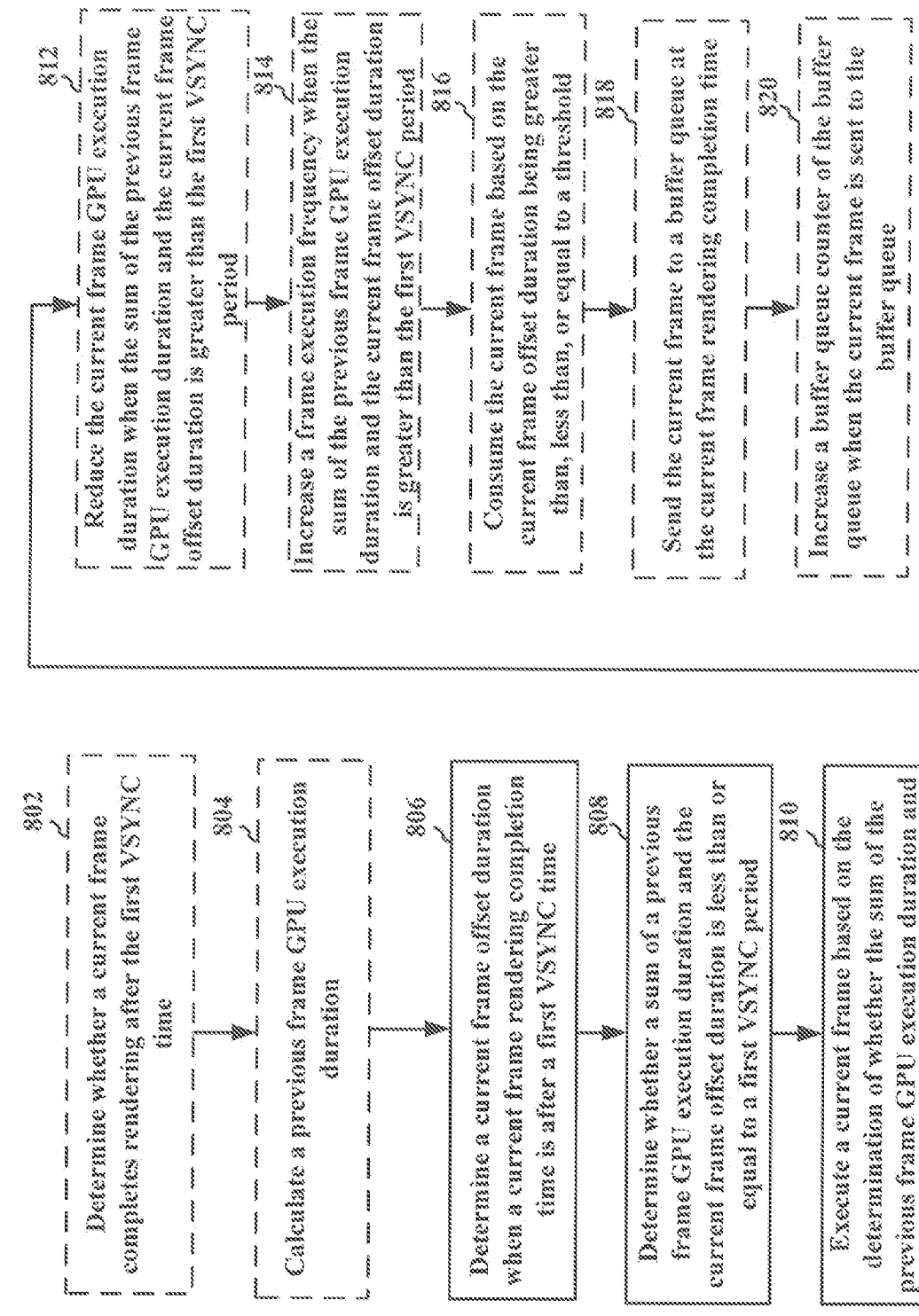
800
802
Determine whether a current frame completes rendering after the first VSYNC time
804
Calculate a previous frame GPU execution duration
806
Determine a current frame offset duration when a current frame rendering completion time is after a first VSYNC time
808
Determine whether a sum of a previous frame GPU execution duration and the current frame offset duration is less than or equal to a first VSYNC period
810
Execute a current frame based on the determination of whether the sum of the previous frame GPU execution duration and the current frame offset duration is less than or equal to the first VSYNC period
812
Reduce the current frame GPU execution duration when the sum of the previous frame GPU execution duration and the current frame offset duration is greater than the first VSYNC period
814
Increase a frame execution frequency when the sum of the previous frame GPU execution duration and the current frame offset duration is greater than the first VSYNC period
816
Consume the current frame based on the current frame offset duration being greater than, less than, or equal to a threshold
818
Send the current frame to a buffer queue at the current frame rendering completion time
820
Increase a buffer queue counter of the buffer queue when the current frame is sent to the buffer queue

FIG. 8

INTERNATIONAL SEARCH REPORT

International application No.
PCT/CN2019/094190

A. CLASSIFICATION OF SUBJECT MATTER

G06F 3/14(2006.01)i

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)

G06F3/-

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)

CNABS,CNTXT,CNKI,WPI,SIPOABS:GPU,frame ,rendering,process,frequency,graphics ,vertical synchronization ,VSYNC, current,duration ,previous

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	WO 2018191086 A1 (MICROSOFT TECHNOLOGY LICENSING LLC) 18 October 2018 (2018-10-18) description paragraph [0045] to [0051], Fig. 4	1-30
A	CN 102185999 A (VTRON TECHNOLOGIES Co., LTD.) 14 September 2011 (2011-09-14) the whole document	1-30
A	CN 102004620 A (VTRON TECHNOLOGIES Co., LTD.) 06 April 2011 (2011-04-06) the whole document	1-30

☐ Further documents are listed in the continuation of Box C. ☑ See patent family annex.

* 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
20 February 2020	**24 March 2020**
Name and mailing address of the ISA/CN **National Intellectual Property Administration, PRC** **6, Xitucheng Rd., Jimen Bridge, Haidian District, Beijing 100088** **China** Facsimile No. **(86-10)62019451**	Authorized officer **YU,Chen** Telephone No. **(86-10)62412319**

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.
PCT/CN2019/094190

Patent document cited in search report	Publication date (day/month/year)	Patent family member(s)	Publication date (day/month/year)
WO 2018191086 A1	18 October 2018	CN 110520819 A	29 November 2019
		US 2018300838 A1	18 October 2018
		US 10319065 B2	11 June 2019
		IN 201947035674 A	11 October 2019
CN 102185999 A	14 September 2011	CN 102185999 B	20 March 2013
CN 102004620 A	06 April 2011	CN 102004620 B	09 May 2012