Intel® Open Source HD Graphics, Intel Iris™ Graphics, and Intel Iris™ Pro Graphics

Programmer's Reference Manual

For the 2015 - 2016 Intel Core™ Processors, Celeron™ Processors, and Pentium™ Processors based on the "Skylake" Platform

Volume 1: Preface

May 2016, Revision 1.0

Creative Commons License

You are free to Share - to copy, distribute, display, and perform the work under the following conditions:

- **Attribution.** You must attribute the work in the manner specified by the author or licensor (but not in any way that suggests that they endorse you or your use of the work).
- **No Derivative Works.** You may not alter, transform, or build upon this work.

Notices and Disclaimers

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL® PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

A "Mission Critical Application" is any application in which failure of the Intel Product could result, directly or indirectly, in personal injury or death. SHOULD YOU PURCHASE OR USE INTEL'S PRODUCTS FOR ANY SUCH MISSION CRITICAL APPLICATION, YOU SHALL INDEMNIFY AND HOLD INTEL AND ITS SUBSIDIARIES, SUBCONTRACTORS AND AFFILIATES, AND THE DIRECTORS, OFFICERS, AND EMPLOYEES OF EACH, HARMLESS AGAINST ALL CLAIMS COSTS, DAMAGES, AND EXPENSES AND REASONABLE ATTORNEYS' FEES ARISING OUT OF, DIRECTLY OR INDIRECTLY, ANY CLAIM OF PRODUCT LIABILITY, PERSONAL INJURY, OR DEATH ARISING IN ANY WAY OUT OF SUCH MISSION CRITICAL APPLICATION, WHETHER OR NOT INTEL OR ITS SUBCONTRACTOR WAS NEGLIGENT IN THE DESIGN, MANUFACTURE, OR WARNING OF THE INTEL PRODUCT OR ANY OF ITS PARTS.

Intel may make changes to specifications and product descriptions at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined". Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The information here is subject to change without notice. Do not finalize a design with this information.

The products described in this document may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Implementations of the I2C bus/protocol may require licenses from various entities, including Philips Electronics N.V. and North American Philips Corporation.

Intel and the Intel logo are trademarks of Intel Corporation in the U.S. and other countries.

* Other names and brands may be claimed as the property of others.

Table of Contents

About This Programmer's Reference Manual

This Preface serves as an Introduction to the Programmer's Reference Manual (PRM) for the Graphics Processing Unit (GPU) included on the Skylake series of chips. The PRM includes both narrative content that explains conceptually how the different modules of the GPU work, and a comprehensive Command Reference that describes named memory addresses/registers, commands, structures and enumerations, at the bit-level.

A Public View of the PRM

For each major chip series, Intel publicly releases a version of the PRM for the Open Source development community and developers at large. For those releases, we release the PRM in Portable Document Format (PDF).

In Multiple PDF Files

Due to file size constraints in generating PDF output (the Skylake "Open Source" version of the PRM is roughly 5,000 pages long), the PRM is generated in multiple PDF files. The organization of those files reflects the major components of the Skylake GPU. Both the narrative content and the Command Reference had to be partitioned into multiple PDF files.

Without Cross-book Links

Cross-book hyperlinks could not reasonably be included for the PDF. Instead, those links have been disabled and colored dark red and display in a serif font for easy recognition. This is an example:

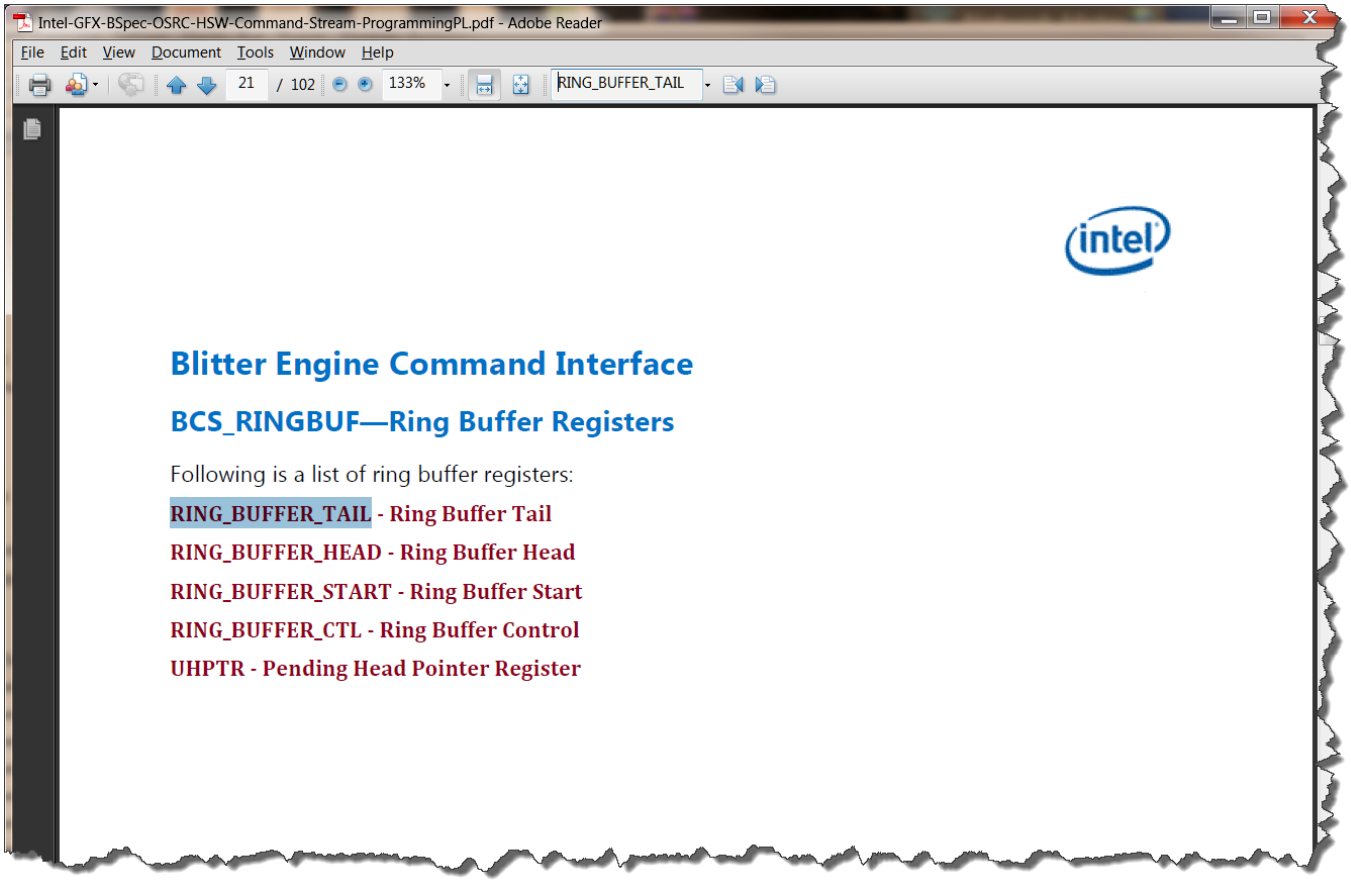
RING_BUFFER_TAIL – Ring Buffer Tail
RING_BUFFER_HEAD – Ring Buffer Head

Searching Across Multiple PDF Files

You can quickly access the detailed explanation for each named memory address, command, structure or enumerator utilizing the Advanced Search capabilities in Acrobat Reader.

To maximize the efficiency of the search, make sure to place all PRM PDFs in a single folder, or at least the four Command Reference PDFs in a single folder.

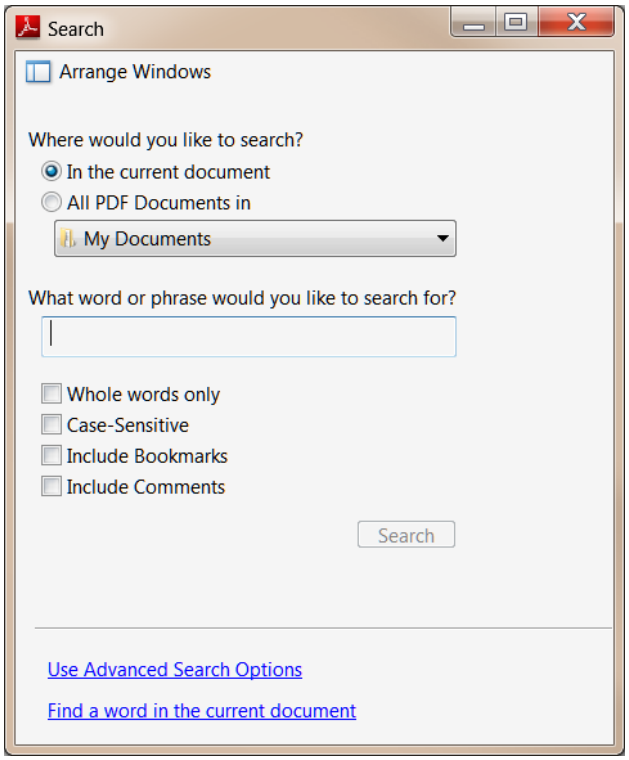
Let's walk through an example PDF search … you might be reading the Command Stream Programming module PDF file and encounter the page illustrated below.



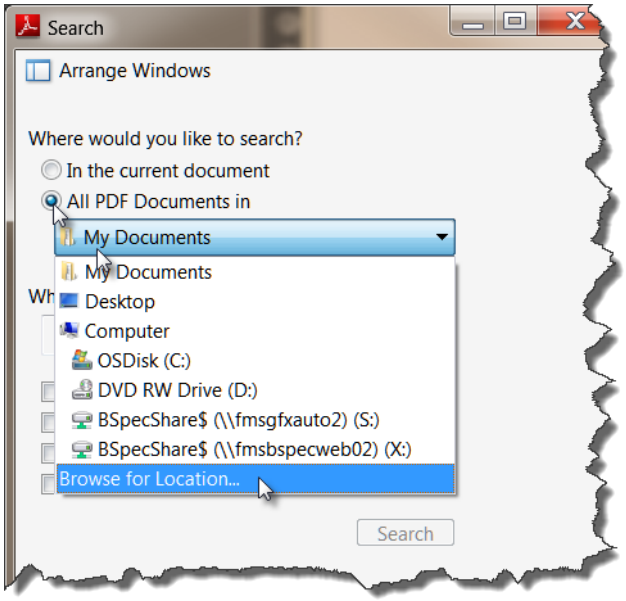
To view the detailed explanation of one of the listed registers … for this example, we will use **RING_BUFFER_TAIL …**

1. Simply select **Search** or **Advanced Search** from the **Edit** menu, depending on the version of Acrobat you might be using. (If you are using some other PDF Reader, it probably will also have advanced search capabilities.)

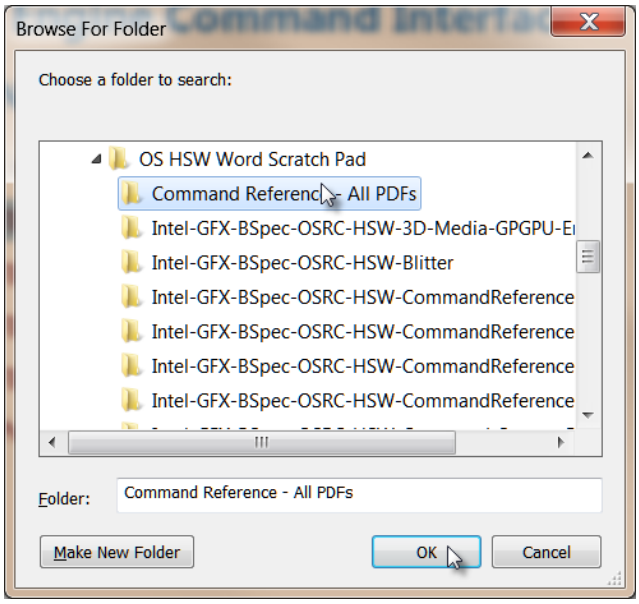
A search dialog box then displays.



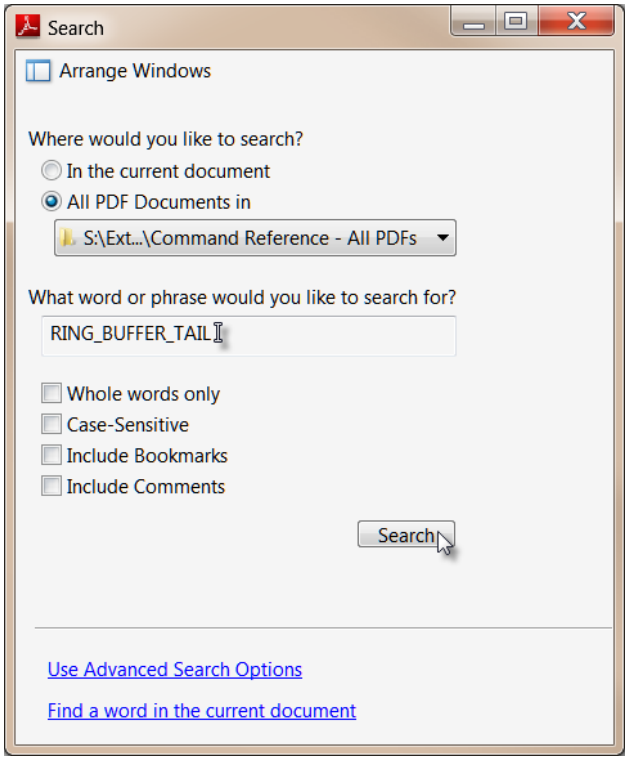
2. Select the **All PDF Documents in** radio button, click the pull-down list box, then select **Browse for Location...**



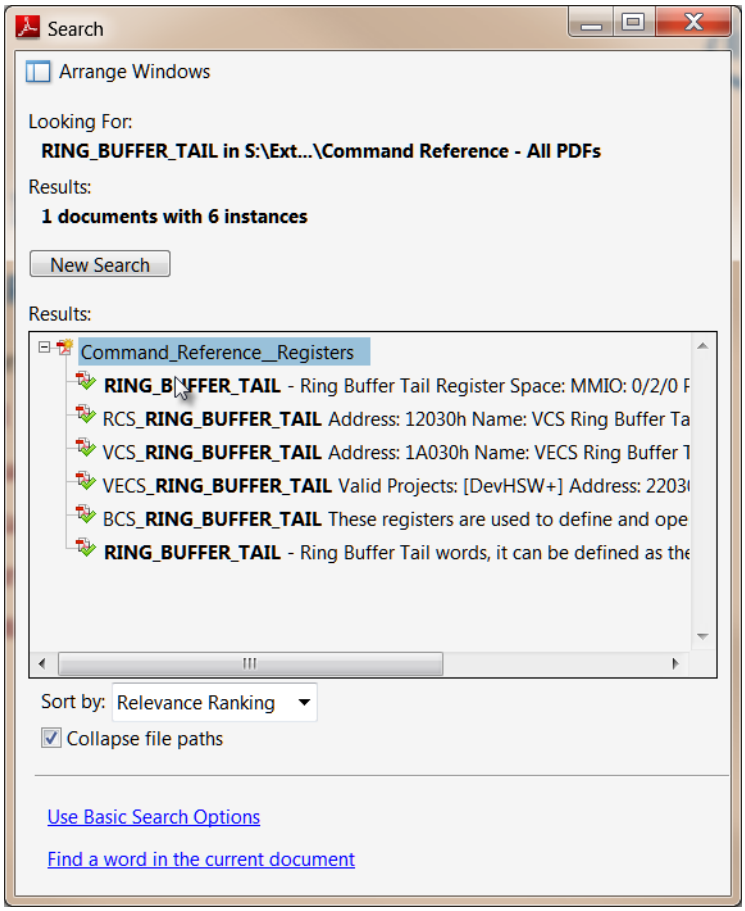
3. Browse to and select the folder containing the pertinent PDF files and click **OK**.



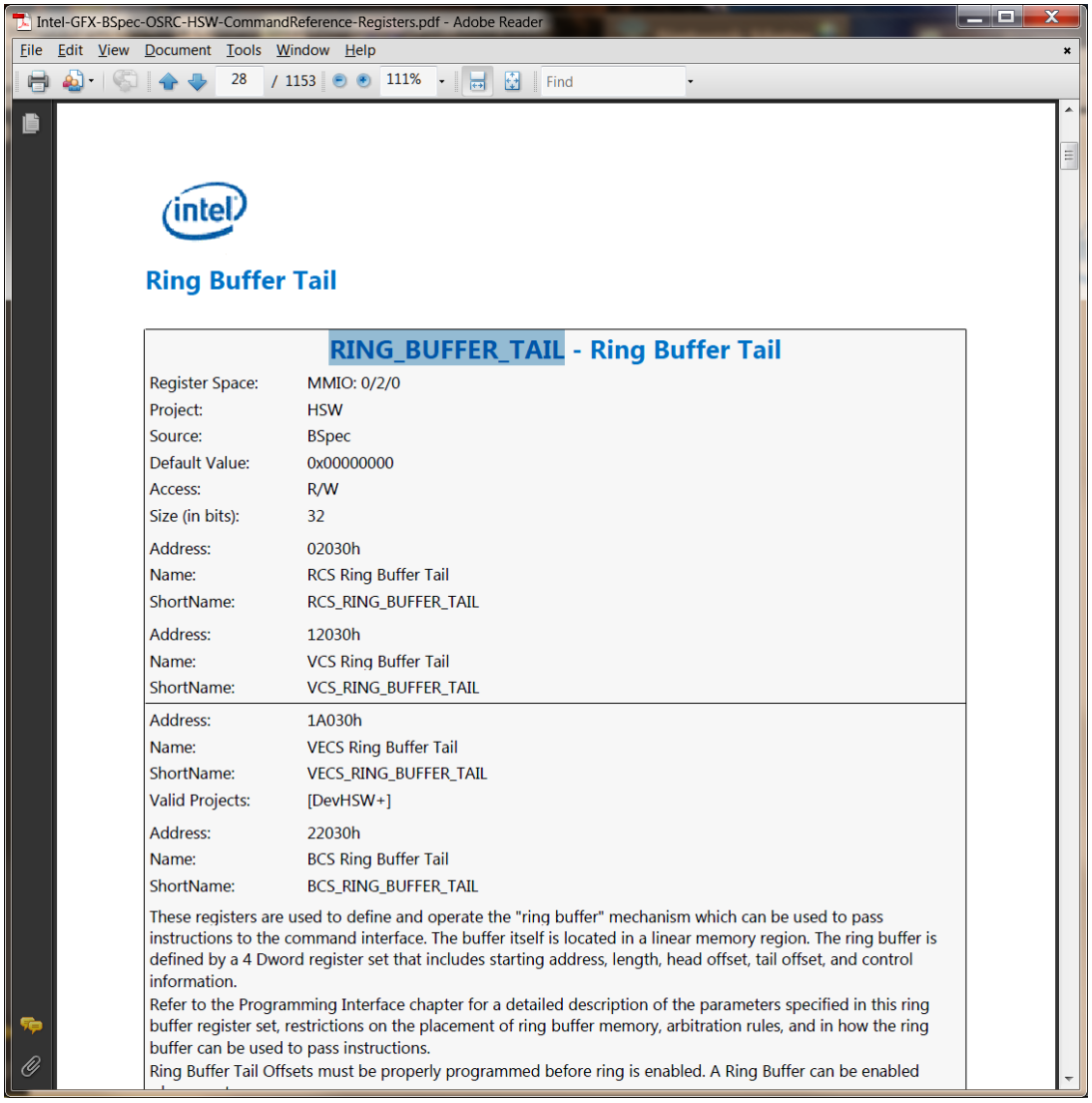
4. The Browse for Folder dialog closes and the focus shifts back to the original Search dialog. Enter the word or phrase from the red, serif text in the **What word or phrase would you like to search for?** text box and click **Search**.



As matches are found, they will be listed. When the search is complete, the dialog will appear similar to the following:



Click the most relevant result and the PDF file containing that result will open at the correct page.



Organization of the PRM

The Intel® Graphics PRM is organized into the following content areas:

Command Reference provides a tabular graphical interface tool for searching for and displaying command opcodes, registers, structures, and instructions.

GPU Overview introduces the GPU and its subsystems. This includes high level descriptions of its hardware pipelines, command formats, register maps, and supported memory/surface formats.

Configurations tracks the evolution of the graphics architecture and provides generational information for many basic architectural attributes.

Memory Views covers the Graphics Memory Interface, including memory interface functions, tiling, the physical graphics memory, page faults and error handling, memory types, common video and non-video surface formats, and other information related to the memory interface.

Command Stream Programming describes the programming interfaces for Command Streamer, which manages the use of the 3D and Media pipelines. It performs switching between pipelines and forwarding command streams to the currently active pipeline. It also manages the allocation of the Unified Return Buffer (URB) and helps support the Constant URB Entry (CURBE) function. It includes the following topics:

Topic
Command Formats
Blitter Engine Command Interface
Render Engine Command Interface
Video Codec Engine Command Interface
Video Enhancement Engine Command Interface
Preemption
Command Streamer (CS) ALU Programming
Resource Streamer

3D/Media/GPGPU Engine includes the following topics:

- Command stream backend processing
- The Graphics L3 large-storage cache, which also acts as a GFX Unified Return Buffer (URB)
- Shared Local Memory (SLM)
- Messaging
- Shared functions for the 3D pipeline
- 3D fixed functions
- Media/GPGPU Pipeline processing via both fixed functions and programmable GPU cores
- Execution units

Media VDBOX covers the **Multi-Format Codec (MFX) Engine**, the hardware fixed-function pipeline that includes both multi-format decoding (MFD) and multi-format encoding (MFC).

Media VEBOX discusses the Video Enhancement pipeline, an independent pipe that manages image enhancement functions. Topics include denoise filtering, deinterlacing, color processing, auto contrast enhancement, the capture pipe, output statistics, and video enhancement engine functions.

HEVC covers the High Efficiency Video Coding (HEVC) Codec Pipeline (HCP), which is a fixed function hardware video codec pipeline responsible for encoding and decoding HEVC streams, with a target resolution of 4kx2k at 60 frames per second.

Blitter discusses the **BL**ock **I**mage **T**ransfer**er** Engine for 2D graphics processing.

Display describes display engine registers, including display HD audio configuration and programming. It includes VGA and extended VGA registers, CPU display registers, North display engine registers, and South display engine registers. The Display section explains audio codec verbs, display audio configuration registers, and the display audio controller registers.

Memory Mapped I/O (**MMIO**) includes information on Slice Registers and Die Recovery Units including Start and End Range offsets.

SFC discusses the commands sent to the **Scalar and Format Converter (SFC)** pipeline from the Media VEBOX and Media VDBOX.

Observability Performance Counters discusses performance visibility, including performance event

Workarounds provide software workarounds for known issues with the GPU.

Device Tags and Definitions

The following table lists device "tags" (abbreviations, projects) used in various parts of this document as aliases for the device names. PRM content without any device tagging applies to all devices in the current spec.

Program Name	Graphics Architecture	Description
Skylake	Gen9	Successor to the HSW/BDW microarchitecture, using a 14nm process.

Reserved Bits and Software Compatibility

In many register, instruction, and memory layout descriptions, certain bits are marked as “Reserved.” When bits are marked as reserved, it is essential for compatibility with future devices that the software treat these bits as having a future, though unknown, effect. The behavior of reserved bits should be regarded as undefined *and unpredictable*. Software should follow the guidelines listed below in dealing with reserved bits:

1. Do not depend on the states of any reserved bits when testing values of registers that contain such bits.
2. Mask out the reserved bits before testing.
3. Do not depend on the states of any reserved bits when storing to an instruction or to a register.
4. When loading a register or formatting an instruction, always load the reserved bits with the values indicated in the documentation (if any), or reload them with the values previously read from the register.

Terminology

Term	Abbr.	Definition
3D Pipeline	--	One of the two pipelines supported in the GPE. The 3D Pipeline is a set of fixed-function units arranged in a pipelined fashion, which process 3D-related commands by spawning EU threads. Typically this processing includes rendering primitives. See *3D Pipeline*.
Adjacency	--	One can consider a single line object as existing in a strip of connected lines. The neighboring line objects are called "adjacent objects", with the non-shared endpoints called the "adjacent vertices." The same concept can be applied to a single triangle object, considering it as existing in a mesh of connected triangles. Each triangle shares edges with three other adjacent triangles, each defined by a non-shared adjacent vertex. Knowledge of these adjacent objects/vertices is required by some object processing algorithms (e.g., silhouette edge detection). See *3D Pipeline*.
Application IP	AIP	Application Instruction Pointer. This is part of the control registers for exception handling for a thread. Upon an exception, hardware moves the current IP into this register and then jumps to System Instruction Pointer (SIP).
Architectural Register File	ARF	A collection of architecturally visible registers for a thread such as address registers, accumulator, flags, notification registers, IP, null, etc.
Binding Table	--	Memory-resident list of pointers to surface state blocks (also in memory).
Binding Table Pointer	BTP	Pointer to a binding table, specified as an offset from the Surface State Base Address register.
Blitter	BLT	Block Image Transferrer
Bypass Mode	--	Mode where a given fixed function (FF) unit is disabled and forwards data down the pipeline unchanged. Not supported by all FF units.
Byte	B	A numerical data type of 8 bits, B represents a signed byte integer.
Child Thread	--	A branch-node or a leaf-node thread that is created by another thread. It is a kind of thread associated with the media fixed function pipeline. A child thread is originated from a thread (the parent) executing on an EU and forwarded to the Thread Dispatcher by the TS unit. A child thread may or may not have child threads depending on whether it is a branch-node or a leaf-node thread. All pre-allocated resources such as URB and scratch memory for a child thread are managed by its parent thread. See also *Parent Thread*.
Clip Space	--	A 4-dimensional coordinate system within which a clipping frustum is defined. Object positions are projected from Clip Space to NDC space via "perspecitive divide" by the W coordinate, and then viewport mapped into Screen Space.
Clipper	--	3D fixed function unit that removes invisible portions of the drawing sequence by discarding (culling) primitives or by "replacing" primitives with one or more primitives that replicate only the visible portion of the original primitive.
Color Calculator	CC	Part of the Data Port shared function, the color calculator performs fixed-function pixel operations (e.g., blending) before writing a result pixel into the render cache.
Command	--	Directive fetched from a ring buffer in memory by the Command Streamer and

Term	Abbr.	Definition
		routed down a pipeline. Should not be confused with instructions which are fetched by the instruction cache subsystem and executed on an EU.
Command Streamer	CS or CSI	Functional unit of the Graphics Processing Engine that fetches commands, parses them, and routes them to the appropriate pipeline.
Constant URB Entry	CURBE	A UE that contains “constant” data for use by various stages of the pipeline.
Control Register	CR	The read-write registers are used for thread mode control and exception handling for a thread.
Core	--	Alternative name for an EU in the multi-processor system. See *EU*.
Data Port	DP	Shared function unit that performs a majority of the memory access types on behalf of SNB+ programs. The Data Port contains the render cache and the constant cache and performs all memory accesses requested by SNB+ programs except those performed by the Sampler. See DataPort.
Degenerate Object	--	Object that is invisible due to coincident vertices or because it does not intersect any sample points (usually due to being tiny or a very thin sliver).
Destination	--	Describes an output or write operand.
Destination Size	--	The number of data elements in the destination of a SIMD instruction.
Destination Width	--	The size of each of (possibly) many elements of the destination of a SIMD instruction.
Double Quad word (DQword)	DQ	A fundamental data type, DQ represents 16 bytes.
Double word (DWord)	D or DW	A fundamental data type, D or DW represents 4 bytes.
Drawing Rectangle	--	A screen-space rectangle within which 3D primitives are rendered. An object's screen-space positions are relative to the Drawing Rectangle origin. See *Strips and Fans*.
End of Block	EOB	A 1-bit flag in the non-zero DCT coefficient data structure indicating the end of an 8x8 block in a DCT coefficient data buffer.
End Of Thread	EOT	A message sideband signal on the Output message bus signifying that the message requester thread is terminated. A thread must have at least one SEND instruction with the EOT bit in the message descriptor field set to properly terminate.
Exception	--	Type of (normally rare) interruption to EU execution of a thread's instructions. An exception occurrence causes the EU thread to begin executing the System Routine, which is designed to handle exceptions.
Execution Channel	--	Single lane of a SIMD operand.
Execution Size	ExecSize	Execution Size indicates the number of data elements processed by an SIMD instruction. It is one of the instruction fields and can be changed per instruction.
Execution Unit	EU	An EU is a multi-threaded processor within the multi-processor system. Each EU is a fully-capable processor containing instruction fetch and decode, register files, source operand swizzle and SIMD ALU, etc. An EU is also referred to as a core.
Execution Unit Identifier	EUID	The 4-bit field within a thread state register (SR0) that identifies the row and column location of the EU where a thread is located. A thread can be uniquely

Term	Abbr.	Definition
		identified by the EUID and TID.
Execution Width	ExecWidth	The width of each of several data elements that may be processed by a single SIMD instruction.
Extended Math Unit	EM	A Shared Function that performs more complex math operations on behalf of several EUs.
Fixed Function	FF	Function of the pipeline that is performed by dedicated (vs. programmable) hardware.
Fixed Function ID	FFID	Unique identifier for a fixed function unit.
Gateway	GW	See *Message Gateway*.
General Register File	GRF	Large read/write register file shared by all the EUs for operand sources and destinations. This is the most commonly used read-write register space organized as an array of 256-bit registers for a thread.
General State Base Address	--	The Graphics Address of a block of memory-resident "state data", which includes state blocks, scratch space, constant buffers, and kernel programs. The contents of this memory block are referenced via offsets from the contents of the General State Base Address register. See *Graphics Processing Engine*.
Geometry Shader	GS	Fixed-function unit between the vertex shader and the clipper that (if enabled) dispatches "geometry shader" threads on its input primitives. Application-supplied geometry shaders normally expand each input primitive into several output primitives to perform 3D modeling algorithms such as fur/fins.
Graphics Address	--	The GPE virtual address of some memory-resident object. This virtual address gets mapped by a GTT or PGTT to a physical memory address. Note that many memory-resident objects are referenced not with Graphics Addresses, but instead with offsets from a "base address register".
Graphics Processing Engine	GPE	Collective name for the Subsystem, the 3D and Media pipelines, and the Command Streamer.
Guardband	GB	Region that may be clipped against to make sure objects do not exceed the limitations of the renderer's coordinate space.
Horizontal Stride	HorzStride	The distance in element-sized units between adjacent elements of a region-based GRF access.
Immediate floating point vector	VF	A numerical data type of 32 bits, an immediate floating point vector of type VF contains 4 floating point elements with 8 bits each. The 8-bit floating point element contains a sign field, a 3-bit exponent field and a 4-bit mantissa field. It may be used to specify the type of an immediate operand in an instruction.
Immediate integer vector	V	A numerical data type of 32 bits, an immediate integer vector of type V contains 8 signed integer elements with 4 bits each. The 4-bit integer element is in 2's complement form. It may be used to specify the type of an immediate operand in an instruction.
Index Buffer	IB	Buffer in memory containing vertex indices.
Intel Architecture	IA	
Instance	--	In the context of the VF unit, an instance is one of a sequence of sets of similar primitive data. Each set has identical vertex data but may have unique instance

Term	Abbr.	Definition
		data that differentiates it from other sets in the sequence.
Instruction	--	Data in memory directing an EU operation. Instructions are fetched from memory, stored in a cache, and executed on one or more cores. Not to be confused with commands which are fetched and parsed by the command streamer and dispatched down the 3D or Media pipeline.
Instruction Pointer	IP	The address (really an offset) of the instruction currently being fetched by an EU. Each EU has its own IP.
Instruction Set Architecture	ISA	The ISA describes the instructions supported by an EU.
Instruction State Cache	ISC	On-chip memory that holds recently-used instructions and state variable values.
Interface Descriptor	--	Media analog of a State Descriptor.
Intermediate Z	IZ	Completion of the Z (depth) test at the front end of the Windower/Masker unit when certain conditions are met (no alpha, no pixel-shader computed Z values, etc.).
Inverse Discrete Cosine Transform	IDCT	The stage in the video decoding pipe between IQ and MC.
Inverse Quantization	IQ	A stage in the video decoding pipe between IS and IDCT.
JIT	--	Just-in-time compiler, aka "jitter".
Kernel	--	A sequence of instructions that is logically part of the driver or generated by the jitter. Differentiated from a Shader which is an application supplied program that is translated by the jitter to instructions.
Logical Ring Context Area	LRCA	Memory area used to store contents of registers and state information required for initiating and resuming communication between software application and hardware graphics pipeline via Ring Buffers.
Least Significant Bit	LSB	The bit with the lowest bit position within a group of bits, which could be a bit group, DWord, field, instruction, memory range, register, or structure. For example, bit 0 of a DWord.
MathBox	--	See *Extended Math Unit*
Media	--	Term for operations such as video decode and encode that are normally performed by the Media pipeline.
Media Pipeline	--	Fixed function stages dedicated to media and “generic” processing, sometimes referred to as the generic pipeline.
Memory-mapped Input/Output	MMIO	A methord for performing input/output between the CPU/GPU and peripheral devices.
Message	--	Messages are data packages transmitted from a thread to another thread, another shared function, or another fixed function. Message passing is the primary communication mechanism of the architecture.
Message Gateway	--	Shared function that enables thread-to-thread message communication/synchronization, used solely by the Media pipeline.
Most Significant Bit	MSB	The bit with the highest bit position within a group of bits, which could be a bit group, DWord, field, instruction, memory range, register, or structure. For

Term	Abbr.	Definition
		example, bit 31 of a DWord.
Motion Compensation	MC	Part of the video decoding pipe.
Motion Picture Expert Group	MPEG	MPEG is the international standard body JTC1/SC29/WG11 under ISO/IEC that has defined audio and video compression standards such as MPEG-1, MPEG-2, and MPEG-4.
Motion Vector Field Selection	MVFS	A four-bit field selecting reference fields for the motion vectors of the current macroblock.
Multiple Render Targets	MRT	Multiple independent surfaces that may be the target of a sequence of 3D or Media commands that use the same surface state.
Normalized Device Coordinates	NDC	Clip Space Coordinates that have been divided by the Clip Space “W” component.
Object	--	A single triangle, line, or point.
OpenGL	OGL	**O**pen **G**raphics **L**ibrary. A graphics API specification associated with Linux.
Parent Thread	--	A thread corresponding to a root-node or a branch-node in thread generation hierarchy. A parent thread may be a root thread or a child thread depending on its position in the thread generation hierarchy.
Pipeline Stage	--	An abstracted element of the 3D Pipeline, providing functions performed by a combination of the corresponding hardware FF unit and the threads spawned by that FF unit.
Pipelined State Pointers	PSP	Pointers to state blocks in memory that are passed down the pipeline.
Pixel Shader	PS	Shader that is supplied by the application, translated by the jitter and is dispatched to the EU by the Windower (conceptually) once per pixel.
Point	--	A drawing object characterized only by position coordinates and width.
Primitive	--	Synonym for object: triangle, rectangle, line, or point.
Primitive Topology	--	A composite primitive such as a triangle strip or a line list. Also includes the objects triangle, line, and point as degenerate cases.
Provoking Vertex	--	The vertex of a primitive topology from which vertex attributes that are constant across the primitive are taken.
Quad Quad word (QQword)	QQ	A fundamental data type, QQ represents 32 bytes.
Quad Word (QWord)	QW	A fundamental data type, QW represents 8 bytes.
Rasterization	--	Conversion of an object represented by vertices into the set of pixels that make up the object.
Region-based addressing	--	Collective term for the register addressing modes available in the EU instruction set that permit discontiguous register data to be fetched and used as a single operand.
Render Cache	RC	Cache in which pixel color and depth information is written before being written to memory, and where prior pixel destination attributes are read in preparation

Term	Abbr.	Definition
		for blending and Z test.
Render Target	RT	A destination surface in memory where render results are written.
Render Target Array Index	--	Selector of which of several render targets the current operation is targeting.
Resource Streamer	RS	Functional unit of the Graphics Processing Engine that examines the commands in the ring buffer in an attempt to pre-process certain long latency items for the remainder of the graphics processing.
Root Thread	--	A root-node thread. A thread corresponds to a root-node in a thread generation hierarchy. It is a kind of thread associated with the media fixed function pipeline. A root thread is originated from the VFE unit and forwarded to the Thread Dispatcher by the TS unit. A root thread may or may not have child threads. A root thread may have scratch memory managed by TS. A root thread with children has its URB resource managed by the VFE.
Sampler	--	Shared function that samples textures and reads data from buffers on behalf of EU programs.
Scratch Space	--	Memory allocated to the subsystem that is used by EU threads for data storage that exceeds their register allocation, persistent storage, storage of mask stack entries beyond the first 16, etc.
Shader	--	A program supplied by the application in a high level shader language, and translated to instructions by the jitter.
Shared Function	SF	Function unit that is shared by EUs. EUs send messages to shared functions, that consume the data and may return results. The Sampler, Data Port, and Extended Math unit are all shared functions.
Shared Function ID	SFID	Unique identifier used by kernels and shaders to target shared functions and to identify their returned messages.
Single Instruction Multiple Data	SIMD	A parallel processing architecture that exploits data parallelism at the instruction level. It can also be used to describe the instructions in such an architecture or to describe the amount of data parallelism in a particular instruction (SIMD8 for example).
Source	--	Describes an input or read operand.
Spawn	--	To initiate a thread for execution on an EU. Done by the thread spawner as well as most FF units in the 3D Pipeline.
Sprite Point	--	Point object using full range texture coordinates. Points that are not sprite points use the texture coordinates of the point's center across the entire point object.
State Descriptor	--	Blocks in memory that describe the state associated with a particular FF, including its associated kernel pointer, kernel resource allowances, and a pointer to its surface state.
State Register	SR	The read-only registers containing the state information of the current thread, including the EUID/TID, Dispatcher Mask, and System IP.
State Variable	SV	An individual state element that can be varied to change the way given primitives are rendered or media objects processed. On state variables persist only in memory and are cached as needed by rendering/processing operations except

Term	Abbr.	Definition
		for a small amount of non-pipelined state.
Stream Output	--	A term for writing the output of a FF unit directly to a memory buffer instead of, or in addition to, the output passing to the next FF unit in the pipeline. Currently only supported for the Geometry Shader (GS) FF unit.
Strips and Fans	SF	Fixed function unit whose main function is to decompose primitive topologies such as strips and fans into primitives or objects.
Sub-Register	--	Subfield of a SIMD register. A SIMD register is an aligned fixed size register for a register file or a register type. For example, a GRF register, *r2*, is a 256-bits wide, 256-bit aligned register. A sub-register, *r2.3:d*, is the fourth dword of GRF register *r2*.
Subsystem	--	The name given to the resources shared by the FF units, including shared functions and EUs.
Surface	--	A rendering operand or destination, including textures, buffers, and render targets.
Surface State Base Pointer	--	Base address used when referencing binding table and surface state data.
Synchronized Root Thread	--	A root thread that is dispatched by TS upon a 'dispatch root thread' message.
System IP	SIP	There is one global System IP register for all the threads. From a thread's point of view, this is a virtual read only register. Upon an exception, hardware performs some bookkeeping and then jumps to SIP.
System Routine	--	Sequence of instructions that handles exceptions. SIP is programmed to point to this routine, and all threads encountering an exception will call it.
Thread	--	An instance of a kernel program executed on an EU. The life cycle for a thread starts from the executing the first instruction after being dispatched from Thread Dispatcher to an EU to the execution of the last instruction – a *send* instruction with EOT that signals the thread termination. Threads in the system may be independent from each other or communicate with each other through Message Gateway share function.
Thread Dispatcher	TD	Functional unit that arbitrates thread initiation requests from Fixed Functions units and instantiates the threads on EUs.
Thread Identifier	TID	The field within a thread state register (SR0) that identifies which thread slots on an EU a thread occupies. A thread can be uniquely identified by the EUID and TID.
Thread Payload	--	Before a thread starting execution, some amount of data is pre-loaded into the thread's GRF (starting at r0). This data is typically a combination of control information provided by the spawning entity (FF Unit) and data read from the URB.
Thread Spawner	TS	The second and the last fixed function stage of the media pipeline that initiates new threads on behalf of generic/media processing.
Topology	--	See *Primitive Topology*.
Unified Return Buffer	URB	The on-chip memory managed/shared by Fixed Functions in order for a thread to return data that will be consumed either by a Fixed Function or other threads.

Term	Abbr.	Definition
Unsigned Byte integer	UB	A numerical data type of 8 bits.
Unsigned Double Word integer	UD	A numerical data type of 32 bits. It may be used to specify the type of an operand in an instruction.
Unsigned Word integer	UW	A numerical data type of 16 bits. It may be used to specify the type of an operand in an instruction.
Unsynchronized Root Thread	--	A root thread that is automatically dispatched by TS.
URB Dereference	--	
URB Entry	UE	URB Entry: A logical entity stored in the URB (such as a vertex), referenced via a URB Handle.
URB Entry Allocation Size	--	Number of URB entries allocated to a Fixed Function unit.
URB Fence	Fence	Virtual, movable boundaries between the URB regions owned by each FF unit.
URB Handle	--	A unique identifier for an URB entry that is passed down a pipeline.
URB Reference	--	
Variable Length Decode	VLD	The first stage of the video decoding pipe that consists mainly of bit-wide operations.
Vertex Buffer	VB	Buffer in memory containing vertex attributes.
Vertex Cache	VC	Cache of Vertex URB Entry (VUE) handles tagged with vertex indices. See the VS chapter for details on this cache.
Vertex Fetcher	VF	The first FF unit in the 3D Pipeline responsible for fetching vertex data from memory. Sometimes referred to as the Vertex Formatter.
Vertex Header	--	Vertex data required for every vertex appearing at the beginning of a Vertex URB Entry.
Vertex ID	--	Unique ID for each vertex that can optionally be included in vertex attribute data sent down the pipeline and used by kernel/shader threads.
Vertex Shader	VS	An API-supplied program that calculates vertex attributes. Also refers to the FF unit that dispatches threads to “shade” (calculate attributes for) vertices.
Vertex URB Entry	VUE	An URB entry that contains data for a specific vertex.
Vertical Stride	VertStride	The distance in element-sized units between 2 vertically-adjacent elements of a region-based GRF access.
Video Front End	VFE	The first fixed function in the generic pipeline; performs fixed-function media operations.
Viewport	VP	
Windower IZ	WIZ	Term for Windower/Masker that encapsulates its early (“intermediate”) depth test function.
Windower/Masker	WM	Fixed function triangle/line rasterizer.
Word	W	A numerical data type of 16 bits, W represents a signed word integer.