# drm/i915 Intel GFX Driver

The drm/i915 driver supports all (with the exception of some very early models) integrated GFX chipsets with both Intel display and rendering blocks. This excludes a set of SoC platforms with an SGX rendering unit, those have basic support through the gma500 drm driver.

## Core Driver Infrastructure

This section covers core driver infrastructure used by both the display and the GEM parts of the driver.

### Runtime Power Management

**System Message: ERROR/3** (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\gpu\(linux-master)(Documentation)(gpu)i915.rst`, **line 19**)

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/intel_runtime_pm.c
   :doc: runtime pm
```

**System Message: ERROR/3** (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\gpu\(linux-master)(Documentation)(gpu)i915.rst`, **line 22**)

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/intel_runtime_pm.c
   :internal:
```

**System Message: ERROR/3** (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\gpu\(linux-master)(Documentation)(gpu)i915.rst`, **line 25**)

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/intel_uncore.c
   :internal:
```

### Interrupt Handling

**System Message: ERROR/3** (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\gpu\(linux-master)(Documentation)(gpu)i915.rst`, **line 31**)

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/i915_irq.c
   :doc: interrupt handling
```

**System Message: ERROR/3** (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\gpu\(linux-master)(Documentation)(gpu)i915.rst`, **line 34**)

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/i915_irq.c
   :functions: intel_irq_init intel_irq_init_hw intel_hpd_init
```

**System Message: ERROR/3** (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\gpu\(linux-master)(Documentation)(gpu)i915.rst`, **line 37**)

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/i915_irq.c
   :functions: intel_runtime_pm_disable_interrupts
```

### Intel GVT-g Guest Support(vGPU)

### Intel GVT-g Host Support(vGPU device model)

### Workarounds

# Display Hardware Handling

This section covers everything related to the display hardware including the mode setting infrastructure, plane, sprite and cursor handling and display, output probing and related topics.

## Mode Setting Infrastructure

The i915 driver is thus far the only DRM driver which doesn't use the common DRM helper code to implement mode setting sequences. Thus it has its own tailor-made infrastructure for executing a display configuration change.

## Frontbuffer Tracking

## Display FIFO Underrun Reporting

## Plane Configuration

This section covers plane configuration and composition with the primary plane, sprites, cursors and overlays. This includes the infrastructure to do atomic vsync'ed updates of all this state and also tightly coupled topics like watermark setup and computation, framebuffer compression and panel self refresh.

## Atomic Plane Helpers

### Asynchronous Page Flip

### Output Probing

This section covers output probing and related infrastructure like the hotplug interrupt storm detection and mitigation code. Note that the i915 driver still uses most of the common DRM helper code for output probing, so those sections fully apply.

### Hotplug

### High Definition Audio

### Intel HDMI LPE Audio Support

```
.. kernel-doc:: drivers/gpu/drm/i915/display/intel_lpe_audio.c
   :doc: LPE Audio integration for HDMI or DP playback
```

```
.. kernel-doc:: drivers/gpu/drm/i915/display/intel_lpe_audio.c
   :internal:
```

## Panel Self Refresh PSR (PSR/SRD)

```
.. kernel-doc:: drivers/gpu/drm/i915/display/intel_psr.c
   :doc: Panel Self Refresh (PSR/SRD)
```

```
.. kernel-doc:: drivers/gpu/drm/i915/display/intel_psr.c
   :internal:
```

## Frame Buffer Compression (FBC)

```
.. kernel-doc:: drivers/gpu/drm/i915/display/intel_fbc.c
   :doc: Frame Buffer Compression (FBC)
```

```
.. kernel-doc:: drivers/gpu/drm/i915/display/intel_fbc.c
   :internal:
```

## Display Refresh Rate Switching (DRRS)

```
.. kernel-doc:: drivers/gpu/drm/i915/display/intel_drrs.c
   :doc: Display Refresh Rate Switching (DRRS)
```

```
.. kernel-doc:: drivers/gpu/drm/i915/display/intel_drrs.c
   :functions: intel_drrs_enable
```

**System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\gpu\(linux-master) (Documentation) (gpu)i915.rst, line 192`)**

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/display/intel_drrs.c
   :functions: intel_drrs_disable
```

**System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\gpu\(linux-master) (Documentation) (gpu)i915.rst, line 195`)**

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/display/intel_drrs.c
   :functions: intel_drrs_invalidate
```

**System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\gpu\(linux-master) (Documentation) (gpu)i915.rst, line 198`)**

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/display/intel_drrs.c
   :functions: intel_drrs_flush
```

**System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\gpu\(linux-master) (Documentation) (gpu)i915.rst, line 201`)**

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/display/intel_drrs.c
   :functions: intel_drrs_init
```

## DPIO

**System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\gpu\(linux-master) (Documentation) (gpu)i915.rst, line 207`)**

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/display/intel_dpio_phy.c
   :doc: DPIO
```

## DMC Firmware Support

**System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\gpu\(linux-master) (Documentation) (gpu)i915.rst, line 213`)**

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/display/intel_dmc.c
   :doc: DMC Firmware Support
```

**System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\gpu\(linux-master) (Documentation) (gpu)i915.rst, line 216`)**

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/display/intel_dmc.c
   :internal:
```

## Video BIOS Table (VBT)

**System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\gpu\(linux-master) (Documentation) (gpu)i915.rst, line 222`)**

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/display/intel_bios.c
   :doc: Video BIOS Table (VBT)
```

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/display/intel_bios.c
   :internal:
```

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/display/intel_vbt_defs.h
   :internal:
```

## Display clocks

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/display/intel_cdclk.c
   :doc: CDCLK / RAWCLK
```

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/display/intel_cdclk.c
   :internal:
```

## Display PLLs

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/display/intel_dpll_mgr.c
   :doc: Display PLLs
```

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/display/intel_dpll_mgr.c
   :internal:
```

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/display/intel_dpll_mgr.h
   :internal:
```

### Display State Buffer

# Memory Management and Command Submission

This sections covers all things related to the GEM implementation in the i915 driver.

## Intel GPU Basics

An Intel GPU has multiple engines. There are several engine types.

- RCS engine is for rendering 3D and performing compute, this is named *I915_EXEC_RENDER* in user space.
- BCS is a blitting (copy) engine, this is named *I915_EXEC_BLT* in user space.
- VCS is a video encode and decode engine, this is named *I915_EXEC_BSD* in user space
- VECS is video enhancement engine, this is named *I915_EXEC_VEBOX* in user space.
- The enumeration *I915_EXEC_DEFAULT* does not refer to specific engine; instead it is to be used by user space to specify a default rendering engine (for 3D) that may or may not be the same as RCS.

The Intel GPU family is a family of integrated GPU's using Unified Memory Access. For having the GPU "do work", user space will feed the GPU batch buffers via one of the ioctls *DRM_IOCTL_I915_GEM_EXECBUFFER2* or *DRM_IOCTL_I915_GEM_EXECBUFFER2_WR*. Most such batchbuffers will instruct the GPU to perform work (for example rendering) and that work needs memory from which to read and memory to which to write. All memory is encapsulated within GEM buffer objects (usually created with the ioctl *DRM_IOCTL_I915_GEM_CREATE*). An ioctl providing a batchbuffer for the GPU to create will also list all GEM buffer objects that the batchbuffer reads and/or writes. For implementation details of memory management see GEM BO Management Implementation Details.

The i915 driver allows user space to create a context via the ioctl *DRM_IOCTL_I915_GEM_CONTEXT_CREATE* which is identified by a 32-bit integer. Such a context should be viewed by user-space as -loosely- analogous to the idea of a CPU process of an operating system. The i915 driver guarantees that commands issued to a fixed context are to be executed so that writes of a previously issued command are seen by reads of following commands. Actions issued between different contexts (even if from the same file descriptor) are NOT given that guarantee and the only way to synchronize across contexts (even from the same file descriptor) is through the use of fences. At least as far back as Gen4, also have that a context carries with it a GPU HW context; the HW context is essentially (most of atleast) the state of a GPU. In addition to the ordering guarantees, the kernel will restore GPU state via HW context when commands are issued to a context, this saves user space the need to restore (most of atleast) the GPU state at the start of each batchbuffer. The non-deprecated ioctls to submit batchbuffer work can pass that ID (in the lower bits of drm_i915_gem_execbuffer2::rsvd1) to identify what context to use with the command.

The GPU has its own memory management and address space. The kernel driver maintains the memory translation table for the GPU. For older GPUs (i.e. those before Gen8), there is a single global such translation table, a global Graphics Translation Table (GTT). For newer generation GPUs each context has its own translation table, called Per-Process Graphics Translation Table (PPGTT). Of important note, is that although PPGTT is named per-process it is actually per context. When user space submits a batchbuffer, the kernel walks the list of GEM buffer objects used by the batchbuffer and guarantees that not only is the memory of each such GEM buffer object resident but it is also present in the (PP)GTT. If the GEM buffer object is not yet placed in the (PP)GTT, then it is given an address. Two consequences of this are: the kernel needs to edit the batchbuffer submitted to write the correct value of the GPU address when a GEM BO is assigned a GPU address and the kernel might evict a different GEM BO from the (PP)GTT to make address room for another GEM BO. Consequently, the ioctls submitting a batchbuffer for execution also include a list of all locations within buffers that refer to GPU-addresses so that the kernel can edit the buffer correctly. This process is dubbed relocation.

## Locking Guidelines

**Note**

This is a description of how the locking should be after refactoring is done. Does not necessarily reflect what the locking looks like while WIP.

1. All locking rules and interface contracts with cross-driver interfaces (dma-buf, dma_fence) need to be followed.

2. No struct_mutex anywhere in the code

3. dma_resv will be the outermost lock (when needed) and ww_acquire_ctx is to be hoisted at highest level and passed down within i915_gem_ctx in the call chain

4. While holding lru/memory manager (buddy, drm_mm, whatever) locks system memory allocations are not allowed

   ○ Enforce this by priming lockdep (with fs_reclaim). If we allocate memory while holding these looks we get a rehash of the shrinker vs. struct_mutex saga, and that would be real bad.

5. Do not nest different lru/memory manager locks within each other. Take them in turn to update memory allocations, relying on the object's dma_resv ww_mutex to serialize against other operations.

6. The suggestion for lru/memory managers locks is that they are small enough to be spinlocks.

7. All features need to come with exhaustive kernel selftests and/or IGT tests when appropriate

8. All LMEM uAPI paths need to be fully restartable (_interruptible() for all locks/waits/sleeps)

   ○ Error handling validation through signal injection. Still the best strategy we have for validating GEM uAPI corner cases. Must be excessively used in the IGT, and we need to check that we really have full path coverage of all error cases.
   ○ -EDEADLK handling with ww_mutex

## GEM BO Management Implementation Details

**System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\gpu\(linux-master)(Documentation)(gpu)i915.rst`, line 384)**

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/i915_vma_types.h
   :doc: Virtual Memory Address
```

## Buffer Object Eviction

This section documents the interface functions for evicting buffer objects to make space available in the virtual gpu address spaces. Note that this is mostly orthogonal to shrinking buffer objects caches, which has the goal to make main memory (shared with the gpu through the unified memory architecture) available.

**System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\gpu\(linux-master)(Documentation)(gpu)i915.rst`, line 396)**

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/i915_gem_evict.c
   :internal:
```

## Buffer Object Memory Shrinking

This section documents the interface function for shrinking memory usage of buffer object caches. Shrinking is used to make main memory available. Note that this is mostly orthogonal to evicting buffer objects, which has the goal to make space in gpu virtual address spaces.

**System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\gpu\(linux-master)(Documentation)(gpu)i915.rst`, line 407)**

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/gem/i915_gem_shrinker.c
   :internal:
```

## Batchbuffer Parsing

## User Batchbuffer Execution

## Scheduling

## Logical Rings, Logical Ring Contexts and Execlists

## Global GTT views

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/i915_gem_gtt.c
   :internal:
```

## GTT Fences and Swizzling

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/gt/intel_ggtt_fencing.c
   :internal:
```

### Global GTT Fence Handling

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/gt/intel_ggtt_fencing.c
   :doc: fence register handling
```

### Hardware Tiling and Swizzling Details

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/gt/intel_ggtt_fencing.c
   :doc: tiling swizzling details
```

## Object Tiling IOCTLs

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/gem/i915_gem_tiling.c
   :internal:
```

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/gem/i915_gem_tiling.c
   :doc: buffer object tiling
```

## Protected Objects

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/pxp/intel_pxp.c
   :doc: PXP
```

# Microcontrollers

Starting from gen9, three microcontrollers are available on the HW: the graphics microcontroller (GuC), the HEVC/H.265 microcontroller (HuC) and the display microcontroller (DMC). The driver is responsible for loading the firmwares on the microcontrollers; the GuC and HuC firmwares are transferred to WOPCM using the DMA engine, while the DMC firmware is written through MMIO.

## WOPCM

### WOPCM Layout

## GuC

### GuC Firmware Layout

### GuC Memory Management

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/gt/uc/intel_guc.c
   :functions: intel_guc_allocate_vma
```

**GuC-specific firmware loader**

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/gt/uc/intel_guc_fw.c
   :internal:
```

**GuC-based command submission**

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/gt/uc/intel_guc_submission.c
   :doc: GuC-based command submission
```

**GuC ABI**

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/gt/uc/abi/guc_messages_abi.h
```

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/gt/uc/abi/guc_communication_mmio_abi.h
```

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/gt/uc/abi/guc_communication_ctb_abi.h
```

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/gt/uc/abi/guc_actions_abi.h
```

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/gt/uc/abi/guc_klvs_abi.h
```

### HuC

### HuC Memory Management

### HuC Firmware Layout

The HuC FW layout is the same as the GuC one, see GuC Firmware Layout

### DMC

See DMC Firmware Support

# Tracing

This sections covers all things related to the tracepoints implemented in the i915 driver.

### i915_ppgtt_create and i915_ppgtt_release

### i915_context_create and i915_context_free

# Perf

### Overview

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/i915_perf.c
   :doc: i915 Perf Overview
```

## Comparison with Core Perf

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/i915_perf.c
   :doc: i915 Perf History and Comparison with Core Perf
```

## i915 Driver Entry Points

This section covers the entrypoints exported outside of i915_perf.c to integrate with drm/i915 and to handle the *DRM_I915_PERF_OPEN* ioctl.

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/i915_perf.c
   :functions: i915_perf_init
```

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/i915_perf.c
   :functions: i915_perf_fini
```

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/i915_perf.c
   :functions: i915_perf_register
```

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/i915_perf.c
   :functions: i915_perf_unregister
```

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/i915_perf.c
   :functions: i915_perf_open_ioctl
```

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/i915_perf.c
   :functions: i915_perf_release
```

```
.. kernel-doc:: drivers/gpu/drm/i915/i915_perf.c
   :functions: i915_perf_add_config_ioctl
```

```
.. kernel-doc:: drivers/gpu/drm/i915/i915_perf.c
   :functions: i915_perf_remove_config_ioctl
```

## i915 Perf Stream

This section covers the stream-semantics-agnostic structures and functions for representing an i915 perf stream FD and associated file operations.

```
.. kernel-doc:: drivers/gpu/drm/i915/i915_perf_types.h
   :functions: i915_perf_stream
```

```
.. kernel-doc:: drivers/gpu/drm/i915/i915_perf_types.h
   :functions: i915_perf_stream_ops
```

```
.. kernel-doc:: drivers/gpu/drm/i915/i915_perf.c
   :functions: read_properties_unlocked
```

```
.. kernel-doc:: drivers/gpu/drm/i915/i915_perf.c
   :functions: i915_perf_open_ioctl_locked
```

```
.. kernel-doc:: drivers/gpu/drm/i915/i915_perf.c
   :functions: i915_perf_destroy_locked
```

**System Message: ERROR/3** (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\gpu\(linux-master)(Documentation)(gpu)i915.rst, line 636`)

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/i915_perf.c
   :functions: i915_perf_read
```

**System Message: ERROR/3** (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\gpu\(linux-master)(Documentation)(gpu)i915.rst, line 638`)

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/i915_perf.c
   :functions: i915_perf_ioctl
```

**System Message: ERROR/3** (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\gpu\(linux-master)(Documentation)(gpu)i915.rst, line 640`)

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/i915_perf.c
   :functions: i915_perf_enable_locked
```

**System Message: ERROR/3** (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\gpu\(linux-master)(Documentation)(gpu)i915.rst, line 642`)

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/i915_perf.c
   :functions: i915_perf_disable_locked
```

**System Message: ERROR/3** (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\gpu\(linux-master)(Documentation)(gpu)i915.rst, line 644`)

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/i915_perf.c
   :functions: i915_perf_poll
```

**System Message: ERROR/3** (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\gpu\(linux-master)(Documentation)(gpu)i915.rst, line 646`)

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/i915_perf.c
   :functions: i915_perf_poll_locked
```

## i915 Perf Observation Architecture Stream

**System Message: ERROR/3** (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\gpu\(linux-master)(Documentation)(gpu)i915.rst, line 652`)

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/i915_perf_types.h
   :functions: i915_oa_ops
```

**System Message: ERROR/3** (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\gpu\(linux-master)(Documentation)(gpu)i915.rst, line 655`)

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/i915_perf.c
   :functions: i915_oa_stream_init
```

## Other i915 Perf Internals

This section simply includes all other currently documented i915 perf internals, in no particular order, but may include some more minor utilities or platform specific details than found in the more high-level sections.

```
        i915_oa_stream_disable
        i915_oa_wait_unlocked
        i915_oa_poll_wait
```

# Style

The drm/i915 driver codebase has some style rules in addition to (and, in some cases, deviating from) the kernel coding style.

## Register macro definition style

The style guide for `i915_reg.h`.

**System Message: ERROR/3** `(D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\gpu\(linux-master)(Documentation)(gpu)i915.rst, line 710)`

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/gpu/drm/i915/i915_reg.h
   :doc: The i915 register macro definition style guide
```